Structure-preserving machine learning for inverse problems

Ferdia John Sherry

University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Peterhouse July 2021

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. It is not substantially the same as any dissertation that I have submitted, or, is being concurrently submitted for a degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. I further state that no substantial part of my dissertation has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University of similar institution except as declared in the Preface and specified in the text.

Ferdia John Sherry
July 2021

Structure-preserving machine learning for inverse problems

Ferdia John Sherry

Inverse problems naturally arise in many scientific settings, and the study of these problems has been crucial in the development of important technologies such as medical imaging. In inverse problems, the goal is to estimate an underlying ground truth u^*, typically an image, from corresponding measurements y, where u^* and y are related by

$$y = \mathfrak{N}(A(u^*)) \tag{1}$$

for some forward operator A and noise-generating process $\mathfrak{N}$ (both of which are generally assumed to be known). Variational regularisation is a well-established approach that can be used to approximately solve inverse problems such as Problem (1). In this approach an image is reconstructed from measurements y by solving a minimisation problem such as

$$\hat{u} = \operatorname*{argmin}_u d(A(u), y) + \alpha J(u). \tag{2}$$

While this approach has proven very successful, it generally requires the parts that make up the optimisation problem to be carefully chosen, and the optimisation problem may require considerable computational effort to solve. There is an active line of research into overcoming these issues using data-driven approaches, which aim to use multiple instances of data to inform a method that can be used on similar data. In this dissertation we investigate ways in which favourable properties of the variational regularisation approach can be combined with a data-driven approach to solving inverse problems.

In the first chapter of the dissertation, we propose a bilevel optimisation framework that can be used to optimise sampling patterns and regularisation parameters for variational image reconstruction in accelerated magnetic resonance imaging. We use this framework to learn sampling patterns that result in better image reconstructions than standard random variable density sampling patterns that sample with the same rate.

In the second chapter of the dissertation, we study the use of group symmetries in learned reconstruction methods for inverse problems. We show that group invariance of a functional implies that the corresponding proximal operator satisfies a group equivariance property. Applying this idea to model proximal operators as roto-translationally equivariant in an unrolled iterative reconstruction method, we show that reconstruction performance is more

robust when tested on images in orientations not seen during training (compared to similar methods that model proximal operators to just be translationally equivariant) and that good methods can be learned with less training data.

In the final chapter of the dissertation, we propose a ResNet-styled neural network architecture that is provably nonexpansive. This architecture can be thought of as composing discretisations of gradient flows along learnable convex potentials. Appealing to a classical result on the numerical integration of ODEs, we show that constraining the operator norms of the weight operators is sufficient to give nonexpansiveness, and additional analysis in the case that the numerical integrator is the forward Euler method shows that the neural network is an averaged operator. This guarantees that its fixed point iterations are convergent, and makes it a natural candidate for a learned denoiser in a Plug-and-Play approach to solving inverse problems.

Acknowledgements

In the first place, my sincere thanks go out to my supervisors, Carola-Bibiane Schönlieb and Matthias Ehrhardt for their constant encouragement and for their enduring patience as they guided me through the various challenges that are characteristic of becoming an independent researcher. I have had the great privilege of being a member of the Cambridge Image Analysis group, the Cambridge Centre for Analysis and the Cantab Capital Institute for the Mathematics of Information. It has been fascinating to be exposed to the multitude of topics studied in these groups, and I think that this has been an important part of my general education as a mathematician. I am grateful to the members of these groups, in particular those who were in my cohort, for many stimulating discussions. The administrators of these groups, Tessa Blackman, Rachel Furner and Josh Stevens, deserve recognition for their roles in keeping everything running; I always found them very willing to help me whenever I had any organisational questions.

In the second half of 2019, the Newton Institute hosted a programme on structure-preserving numerical methods. This was the start of a fruitful collaboration with Elena Celledoni, Christian Etmann, Robert McLachlan and Brynjulf Owren. I am thankful for the many discussions we had on machine learning, which resulted in us writing a review paper on structure-preserving deep learning (Celledoni et al., 2021a). This work in turn inspired the work presented in Chapter 2 and Chapter 3 of this dissertation. I very much look forward to continuing this collaboration in the future.

Cambridge is not just an inspiring place to study mathematics. I have also had the opportunity to explore a wide variety of extracurricular activities. Most notably for me was that I was able to start learning Irish in Margo Griffin-Wilson's classes at the ASNC Department. This is something I had wanted to do for years, and I had never imagined that Cambridge would be the place where I would get to do so. Besides attending the excellent classes in Cambridge, I was given the chance to study at Coláiste na Rinne in the summer of 2018, which was a wonderful experience. Go raibh míle maith agat, a Margo.

Finally, I need to thank my family for everything they have done for me. All throughout my studies, my parents have supported me at every step that I have taken. I am very appreciative of my aunts, uncles and cousins who have welcomed me into their homes, and have fond memories of the time spent with them during the holidays that I have had in the past few years.

Table of contents

List of figures

List of tables

Introduction

Inverse problems

In this dissertation we will concern ourselves with inverse imaging problems: given indirect measurements y of an underlying ground truth image u^*, the goal is to accurately estimate u^*. Generally, the measurement process is (at least partially) known, and it can be described as

$$y = \mathfrak{N}(A(u^*)), \tag{3}$$

with A a forward operator and $\mathfrak{N}$ a noise-generating process, which usually represents a small, potentially random, perturbation of the identity. This inversion problem is said to be well-posed in the sense of Hadamard (Hadamard, 1902) if a set of three conditions is satisfied:

- **Existence:** For any set of uncorrupted measurements y, there is a solution u with $A(u) = y$,
- **Uniqueness:** If u_1 and u_2 are two solutions corresponding to a set of uncorrupted measurements $y = A(u_1) = A(u_2)$, then $u_1 = u_2$,
- **Continuity:** The solution u depends continuously on the set of measurements y.

If a problem fails to satisfy at least one of these conditions, it is called ill-posed, and most inverse problems of interest turn out to be ill-posed. For many problems of interest, it is possible to define a pseudoinverse that addresses at least the first two conditions for well-posedness. For instance, if $A : \mathcal{X} \to \mathcal{Y}$ is a bounded linear operator between Hilbert spaces, the Moore-Penrose inverse $A^\dagger : \mathrm{dom}(A^\dagger) \to \mathcal{X}$, which is a potentially unbounded operator with $\mathrm{dom}(A^\dagger) = \mathrm{im}(A) \oplus \mathrm{im}(A)^\perp$, fulfills this role (Engl et al., 1996; Moore, 1920; Penrose, 1955). Usually however, the final condition for well-posedness is not satisfied by such a pseudoinverse and regularisation is needed to ensure stable reconstructions in the presence of noise. Even when this condition is satisfied, as in the case of subsampled MRI (corresponding to the

forward operator $A = \mathcal{S}\mathcal{F}$ with $\mathcal{S}$ a binary multiplication operator), the reconstruction given by this specific pseudoinverse may be of an unacceptable quality. This problem too can be solved through judicious application of a regularisation method.

Variational regularisation

Variational regularisation (Engl et al., 1996; Hansen, 2010) is a standard approach to regularising the solution of Problem (3). It proposes to estimate the ground truth image u^* from measurements y by solving a variational problem of the form

$$\hat{u} = \operatorname*{argmin}_u d(A(u), y) + \alpha J(u), \tag{4}$$

where d is a data discrepancy functional, J is a regularisation functional and $\alpha \geqslant 0$ is a regularisation parameter controlling the trade-off between the two terms. Often, the form of the objective function will be chosen based on statistical considerations: if we have a density $y \mapsto p(y|A(u))$ for the likelihood and $u \mapsto p(u)$ for the prior, the posterior distribution has a density $p(u|y) \propto p(y|A(u))p(u)$ by Bayes' theorem. Maximising the posterior is then equivalent to solving Problem (4) with $d(A(u), y) = -\log(p(y|A(u)))$ and $\alpha J(u) = -\log(p(u))$.

One of the main desirable properties that variational regularisation satisfies is that it is provably stable. Let us consider the basic case where $\mathcal{X}$ and $\mathcal{Y}$ are Hilbert spaces, $A : \mathcal{X} \to \mathcal{Y}$ is a bounded linear operator, $J : \mathcal{X} \to \mathbf{R} \cup \{+\infty\}$ is a coercive, proper, convex, weakly lower semi-continuous functional and the data discrepancy functional is the squared norm. The variational regularisation approach defines a family of maps $\{R_\alpha\}_{\alpha>0}$, $R_\alpha : \mathcal{Y} \to \mathcal{X}$ with

$$R_\alpha(y) = \operatorname*{argmin}_{u \in \mathcal{X}} \frac{1}{2}\|Au - y\|^2 + \alpha J(u).$$

The first order optimality conditions defining R_α show that if $u_1 = R_\alpha(y_1)$ and $u_2 = R_\alpha(y_2)$, then there are $p_i \in \partial J(u_i)$ such that

$$A^*(Au_i - y_i) + \alpha p_i = 0.$$

Multiplying both of these equations by $u_1 - u_2$ and rearranging we find

$$\alpha\langle p_1 - p_2, u_1 - u_2\rangle + \|A(u_1 - u_2)\|^2 = \langle y_1 - y_2, A(u_1 - u_2)\rangle \leqslant \|y_1 - y_2\|\|A(u_1 - u_2)\|.$$

Since J is convex, we have $\langle p_1 - p_2, u_1 - u_2 \rangle \geqslant 0$. From this, we find that

$$\|A(u_1 - u_2)\|^2 \leqslant \|y_1 - y_2\| \|A(u_1 - u_2)\|, \quad \text{so that} \quad \|A(u_1 - u_2)\| \leqslant \|y_1 - y_2\|.$$

Substituting this back into the above inequality, we find

$$\alpha \langle p_1 - p_2, u_1 - u_2 \rangle \leqslant \|y_1 - y_2\|^2. \tag{5}$$

If J is strongly convex with constant $\mu > 0$, Inequality (5) immediately gives us that R_α is $(1/\sqrt{\alpha\mu})$-Lipschitz:

$$\|u_1 - u_2\|^2 \leqslant \frac{1}{\alpha\mu} \|y_1 - y_2\|^2.$$

In fact, even if J is not strongly convex, Inequality (5) may still be of interest as a weaker form of stability, especially when it is written in slightly different notation. Recall the definition of the Bregman divergence (Bregman, 1967):

$$D_J^{p_1}(u_1, u_2) = J(u_2) - J(u_1) - \langle p_1, u_2 - u_1 \rangle \geqslant 0,$$

and the corresponding symmetrised Bregman divergence:

$$D_{\text{symm},J}^{p_1,p_2}(u_1, u_2) = D_J^{p_1}(u_1, u_2) + D_J^{p_2}(u_2, u_1) = \langle p_1 - p_2, u_1 - u_2 \rangle.$$

Hence Inequality (5) can be rephrased as

$$D_{\text{symm},J}^{p_1,p_2}(u_1, u_2) \leqslant \frac{1}{\alpha} \|y_1 - y_2\|^2.$$

In addition to this stability result, it can be shown à la Theorem 5.2 in Engl et al. (1996) that the variational regularisation is a convergent regularisation in the sense that it converges to a pseudoinverse: if $y \in \text{im}(A)$, y^δ are such that $\|y^\delta - y\| \leqslant \delta$ and $\alpha(\delta)$ are chosen so that $\alpha(\delta) \to 0$ and $\delta^2/\alpha(\delta) \to 0$ as $\delta \to 0$, then $R_{\alpha(\delta)} y^\delta$ converges weakly to a J-minimising solution of $Au = y$. This J-minimising solution can be thought of as a pseudoinverse since it generalises the well known characterisation of the Moore-Penrose pseudoinverse (Theorem 2.5 in Engl et al. (1996)) as the minimum norm solution.

Both stability and convergence to a reasonable pseudoinverse may be thought of as necessary conditions for a regularisation method to be trustworthy, and they hold with quite generic assumptions on the regularisation functional (at least for linear inverse problems). In practice, however, we are mostly interested in the nonasymptotic setting (i.e. where the noise

level, and correspondingly the effect of the regularisation, is bounded below by a positive amount). It is crucial to understand that we are not solving the original problem in this setting, but a well-posed problem that is in some sense (made precise by the convergence results) close to the original problem. In particular, if we take a statistical perspective of the problem, we naturally encounter a bias-variance trade-off (Hastie et al., 2009): as $\alpha \to \infty$ the estimate produced by solving Problem (4) becomes more biased towards being a minimiser of J and its variance eventually vanishes, whereas when $\alpha \to 0$ the bias disappears but the variance increases (and potentially blows up for an ill-posed inverse problem). Furthermore, whenever we estimate u^* as $\hat{u}$, we can decompose the estimation error into a variance term and a squared-bias term:

$$\begin{aligned}\mathbb{E}\|\hat{u} - u^*\|^2 &= \mathbb{E}\|\hat{u}\|^2 - 2\langle \mathbb{E}[\hat{u}], u^*\rangle + \|u^*\|^2 \\ &= \left(\mathbb{E}\|\hat{u}\|^2 - \|\mathbb{E}[\hat{u}]\|^2\right) + \left(\|\mathbb{E}[\hat{u}]\|^2 - 2\langle \mathbb{E}[\hat{u}], u^*\rangle + \|u^*\|^2\right) \\ &= \mathbb{E}\|\hat{u} - \mathbb{E}[\hat{u}]\|^2 + \|\mathbb{E}[\hat{u}] - u^*\|^2.\end{aligned}$$

Here, the first term on the right-hand side is the variance of $\hat{u}$ and the second term is its squared bias. By the above reasoning, the estimation error plotted against the regularisation parameter α will be U-shaped, and the optimal α can be thought of as making the optimal trade-off between bias and variance of the estimate. In this nonasymptotic setting, the precise choice of regularisation functional and regularisation parameter α needs to be carefully considered, since it can have large effects on the achieved reconstruction quality. Considerable amounts of work have gone into handcrafting regularisation functionals to ensure high quality reconstructions, some standard examples being

- the total variation (TV) functional (Chavent and Kunisch, 1997; Osher et al., 2005; Rudin et al., 1992) and its higher order generalisations (Benning et al., 2013; Bredies and Holler, 2014; Bredies et al., 2010; Hu and Jacob, 2012; Papafitsoros and Schönlieb, 2014; Scherzer, 2007), which promote piecewise smoothness of the recovered images,
- sparsity penalties, which encourage sparsity of the recovered images in certain representations such as wavelets or their generalisations (Candès and Donoho, 1999; Chaux et al., 2007; Guo and Labate, 2007). This approach is supported by the theory of compressed sensing, starting with Candès et al. (2006); Donoho (2006).

Given the difficulties involved in choosing a regularisation functional and regularisation parameter, a natural question is whether it is possible to learn good choices from data. This leads us to the study of data-driven approaches to inverse problems.

Machine learning approaches to inverse problems

Assuming that our variational regularisation problem, Problem (4), contains some parameters that are not fixed a priori, we could endeavour to choose them by solving a bilevel optimisation problem of the following form:

$$\begin{aligned} &\min_{p,\hat{u}} \mathbb{E}[L(\hat{u}, p)] \\ \text{s.t.} \quad &\hat{u} = \operatorname*{argmin}_{u \in X} E(u, y; p). \end{aligned} \tag{6}$$

Here E is the objective function of Problem (4), p represents the free parameters of E and L is a potentially random loss function that penalises low quality reconstructions and undesirable choices of p. The expectation is taken over the joint distribution of ground truth images u and corresponding measurements y, or an empirical approximation to it depending on a finite number of samples. Commonly, L depends on u^*, e.g. $L(\hat{u}) = \|u^* - \hat{u}\|^2/2$, in which case Problem (6) is a supervised learning problem, although this is not strictly necessary. We refer to this approach to learning parts of a variational regularisation problem as bilevel learning (Calatroni et al., 2015; Chen et al., 2014; De los Reyes and Schönlieb, 2013; De los Reyes et al., 2015; Kunisch and Pock, 2013; Samuel and Tappen, 2009).

The bilevel optimisation approach to choosing free parameters of a variational reconstruction has the notable advantage that, after training, the learned reconstruction method is just a variational regularisation method and as such enjoys the guarantees that can be given for these methods. On the other hand, even when taking the potential theoretical difficulties of the nonconvexity of Problem (6) for granted, existing methods for solving bilevel learning problems require considerable amounts of computational effort; iterative methods for solving the bilevel learning problem typically require the variational reconstruction problem to be solved at least once per iteration. Generally, the variational reconstruction problem is solved using an iterative solver too, making each outer iteration of the bilevel optimisation solver expensive.

In recent years deep learning (LeCun et al., 2015) has become one of the most active branches of machine learning research, and it has seen widespread application to other fields of science. Broadly speaking, the philosophy of deep learning is to use extremely flexible neural network models, usually trained with large amounts of data on powerful computing equipment. Deep learning has been used to break old records by significant margins on a wide range of tasks including image classification (Krizhevsky et al., 2012; Szegedy et al., 2015),

playing board games (Silver et al., 2016, 2018), modelling natural language (Brown et al., 2020; Radford et al., 2019) and protein structure prediction (Jumper et al., 2020). Reflection on these successes has led some people to believe that progress in artificial intelligence is ultimately mainly dependent on scaling up general purpose methods such as neural networks; Richard Sutton expounds this view in his famous "Bitter Lesson" (Sutton, 2019):

> "The bitter lesson is based on the historical observations that 1) AI researchers have often tried to build knowledge into their agents, 2) this always helps in the short term, and is personally satisfying to the researcher, but 3) in the long run it plateaus and even inhibits further progress, and 4) breakthrough progress eventually arrives by an opposing approach based on scaling computation by search and learning. The eventual success is tinged with bitterness, and often incompletely digested, because it is success over a favored, human-centric approach."

Early approaches to using deep learning to solve inverse problems have followed this philosophy quite closely, using minimal knowledge of the forward model, and instead exploiting the flexibility of neural networks. Some representative examples include:

- AUTOMAP (Zhu et al., 2018) proposes a neural network architecture Φ that is directly trained on pairs of ground truth images u^* and noisy measurements y to perform the inversion:

  $$\min_{\Phi} \mathbb{E}\|\Phi(y) - u^*\|^2.$$

  After training, we can estimate the underlying image from measurements y by $\hat{u} = \Phi(y)$. Promising results are reported on an MRI problem compared to naive inversion using the Moore-Penrose pseudoinverse.

- The post-processing approach (Jin et al., 2017) assumes that there is a reasonable pseudoinverse $A^\dagger$ and attempts to correct artefacts by training a general purpose neural network for images (the U-net architecture (Ronneberger et al., 2015)) to do so on pairs of ground truth images u^* and noisy measurements y:

  $$\min_{\Phi} \mathbb{E}\|\Phi(A^\dagger(y)) - u^*\|^2.$$

  After training, we can estimate the underlying image from measurements y by $\hat{u} = \Phi(A^\dagger(y))$. The authors report favourable comparisons to TV regularised variational reconstructions in two ways: reconstruction quality is higher and it takes much less time to compute.

Despite these positive results, it is good to take a more nuanced view. The same image classifiers that set the new state of the art have been shown to be vulnerable to adversarial examples (Akhtar and Mian, 2018; Goodfellow et al., 2015): it is possible to ruin the performance of neural network image classifiers by applying visually imperceptible perturbations to the input images before passing them to the network. It has been shown that similar instabilities exist in deep learning methods for inverse problems such as those mentioned above (Antun et al., 2020). This should not come as too much of a surprise considering the effects that ill-posedness may have. When the (pseudo)inverse is not continuous, small variations in measurements can correspond to large variations in estimated images, dooming any attempt to directly solve the original inverse problem such as Problem (3) to being unstable. Furthermore, in a response (Welling, 2019) to Richard Sutton's "Bitter Lesson", Max Welling highlights that many of deep learning's success stories have depended on being able to gather or generate massive amounts of training data, a condition which can not generally be assumed to hold:

> "But from Rich's argumentation there is one really important factor missing: besides compute, data is perhaps the more fundamental raw material of machine learning. All the examples above share one crucial property, namely that they are very well, and rather narrowly defined problems where you can either generate your own data (e.g. alphaGO) or have ample data available (e.g. speech). In these regimes data-driven, discriminative, black box methods such as DL shine. We can view this as interpolation problems. The input domain is well delimited, we have sufficient data to cover that input domain and interpolate between the dots. The trouble starts when we need to extrapolate."

All of this is to say that we should not be dismissive of principled, model-based approaches to designing machine learning methods for inverse problems. The past few years have seen a proliferation of methods for inverse problems that take a model-based approach by combining concepts from variational regularisation and deep learning. Although the boundaries between these groups can be a bit vague, these approaches can be roughly split into categories as follows:

- A regularisation functional can be parametrised by a neural network, and not trained in the end-to-end fashion as in the bilevel learning setting described in Problem (6). This includes the possibility of training a regularisation functional as a critic to discriminate between ground truth images and images generated by a naive inversion (Lunz et al.,

2018; Mukherjee et al., 2021):

$$J := \operatorname*{argmax}_{1\text{-Lipschitz neural networks } \Phi:\mathcal{X}\to\mathbf{R}} \mathbb{E}[\Phi(A^\dagger(y)) - \Phi(u^*)],$$

or training an autoencoder to map a naive inversion to its corresponding artefacts and taking the squared norm of the encoder part as a regularisation functional (Li et al., 2020):

$$J(u) := \|\Phi(u)\|^2, \text{ where } \Phi, \Psi = \operatorname*{argmin}_{\substack{\text{encoders } \Phi:\mathcal{X}\to\mathcal{Z} \\ \text{decoders } \Psi:\mathcal{Z}\to\mathcal{X}}} \mathbb{E}\|\Psi(\Phi(A^\dagger(A(u^*)))) - (A^\dagger(A(u^*)) - u^*)\|^2.$$

Another notable option in this category is regularisation by denoising (Reehorst and Schniter, 2019; Romano et al., 2017), which proposes to take a predetermined denoiser Φ (which could be a neural network) and uses this to define a regularisation functional $J(u) = \langle u, u - \Phi(u)\rangle/2$.

- The Plug-and-Play approach (Chan, 2016; Chan et al., 2016; Sreehari et al., 2016; Venkatakrishnan et al., 2013) starts from an iterative splitting optimisation method for Problem (4), such as ADMM or forward-backward splitting (Parikh and Boyd, 2014). These algorithms contain steps in which the proximal operator (Moreau, 1965) of the regularisation functional J needs to be computed:

$$\operatorname{prox}_{\tau J}(u) := \operatorname*{argmin}_{u'} \frac{1}{2}\|u - u'\|^2 + \tau J(u') = (\mathrm{id} + \tau\partial J)^{-1}(u).$$

This step can be interpreted as a denoising step and the Plug-and-Play approach proposes to solve the inverse problem by replacing it by a high-performance Gaussian denoiser, which may be a neural network. Excellent performance is observed on various inverse problems when using these general-purpose neural network denoisers in a Plug-and-Play method (Meinhardt et al., 2017; Zhang et al., 2017b).

- Learned iterative reconstruction methods (Adler and Öktem, 2017, 2018; Putzky and Welling, 2017) also take inspiration from iterative optimisation methods for solving variational problems such as Problem (4). They deviate from these methods in that they truncate the algorithm used to a small fraction of the number of iterations and replace parts of each iteration by neural networks. Consider for instance the proximal gradient

method (Beck and Teboulle, 2009; Bruck, 1977; Passty, 1979) for Problem (4):

$$\begin{cases} u^0 = 0, \\ u^{i+1} = \mathrm{prox}_{\tau^i J}(u^i - \tau^i \nabla_u d(A(u^i), y)). \end{cases}$$

A natural learned iterative reconstruction method inspired by this method could be

$$\begin{cases} u^0, s^0 = 0, 0, \\ u^{i+1}, s^{i+1} = \widehat{\mathrm{prox}}_i(u^i, \nabla_u d(A(u^i), y), s^i), \\ \Phi(y) := u^{\mathrm{it}}, \end{cases} \tag{7}$$

where $\widehat{\mathrm{prox}}_i$ are learnable neural networks and s^i are auxiliary memory variables. The overall algorithm is trained in an end-to-end fashion to minimise reconstruction error:

$$\min_{\Phi \text{ constrained as in Equation (7)}} \mathbb{E}\|\Phi(y) - u^*\|^2.$$

This paradigm of learned image reconstruction has shown great potential, vastly improving the reconstruction quality and reducing time needed to compute a reconstruction, when compared to traditional variational regularisation approaches. One thing to note, however, is that the stability of the variational regularisation approach does not carry over directly to the learned iterative reconstruction method: the variational network (Hammernik et al., 2018) is a learned iterative reconstruction method that was shown in Antun et al. (2020) to be susceptible to adversarial attacks.

Contributions

This dissertation consists of three main chapters, each of which studies specific ways in which desirable forms of structure can be incorporated into machine learning methods for inverse problems:

1. **Learning the sampling pattern for MRI** is based on a paper published in *IEEE Transactions on Medical Imaging* (Sherry et al., 2020). In this work we propose and study a bilevel learning problem like Problem (6), that can be used to jointly learn an optimal sampling pattern and regularisation parameters for the problem of compressed sensing MRI. This was a collaboration with Martin Benning, Juan Carlos De los Reyes, Matthias Ehrhardt, Martin Graves, Georg Maierhofer, Carola-Bibiane Schönlieb and Guy Williams. This project was initially started as a summer project for Georg in the summer of 2016. After Georg's work on small toy problems using total variation regularisation in the lower level problem, I identified a subtle error in the solution method used for the lower level problem and started from scratch in order to allow for more general regularisation functionals and to scale up to larger images. Martin Graves and Guy provided the MRI data that was used in the experiments. I wrote the paper and ran all the experiments for the paper, while being guided by discussions with my collaborators.

2. **Equivariant neural networks for inverse problems** is based on a paper accepted for publication in *Inverse Problems* (Celledoni et al., 2021b). In this work we study the use of roto-translationally equivariant neural networks as models for proximal operators in a learned iterative reconstruction method. This was a collaboration with Elena Celledoni, Matthias Ehrhardt, Christian Etmann, Brynjulf Owren and Carola-Bibiane Schönlieb. In the process of writing a review paper (Celledoni et al., 2021a), I realised that equivariance could naturally be used in learned reconstruction methods for inverse problems. I wrote the paper and ran all the experiments for the paper, while being guided by discussions with my collaborators.

3. **Nonexpansive neural networks inspired by ODEs and convex analysis** is based on work that I have submitted for publication. In this work we study a class of ResNet style architectures that are provably nonexpansive as long as the operator norms of the learnable weights are constrained appropriately. This was a collaboration with Elena Celledoni, Matthias Ehrhardt, Christian Etmann, Brynjulf Owren and Carola-Bibiane Schönlieb. While writing the aforementioned review paper (Celledoni et al.,

2021a), Brynjulf noted that the nonexpansiveness of flows along vector fields satisfying a certain monotonicity condition is preserved by discretisation using certain Runge-Kutta methods. I made the connection to gradient flows in convex potentials and wrote the paper and ran all the experiments for the paper, while being guided by discussions with my collaborators.

Chapter 1

Learning the sampling pattern for MRI

1.1 Introduction

The field of compressed sensing is founded on the realisation that in inverse problems it is often possible to recover signals from incomplete measurements. To do so, the inherent structure of signals and images is exploited. Finding a sparse representation for the unknown signal reduces the number of unknowns and consequently the number of measurements required for reconstruction. This is of great interest in many applications, where external reasons (such as cost or time constraints) typically imply that one should take as few measurements as are required to obtain an adequate reconstruction. A specific example of such an application is magnetic resonance imaging (MRI). In MRI, measurements are modelled as samples of the Fourier transform (points in so-called k-space) of the signal that is to be recovered and taking measurements is a time-intensive procedure. Keeping acquisition times short is important to ensure patient comfort and to mitigate motion artefacts, and it increases patient throughput, thus making MRI effectively cheaper. Hence, MRI is a natural candidate for the application of compressed sensing methodology. While the first theoretical results of compressed sensing (as in Candès et al. (2006), in which exact recovery results are proven for uniform random sampling strategies) do not apply well to MRI, three underlying principles were identified that enable the success of compressed sensing (Lustig et al., 2007a; Sodickson et al., 2015):

1. sparsity or compressibility of the signal to be recovered (in some sparsifying transform, such as a wavelet transform)

2. incoherent measurements (with respect to the aforementioned sparsifying transform) and

3. a nonlinear reconstruction algorithm that takes advantage of the sparsity structure in the true signal.

The nonlinear reconstruction algorithm often takes the form of a variational regularisation problem:

$$\min_u \frac{1}{2}\|\mathcal{S}\mathcal{F}u - y\|^2 + \alpha J(u), \tag{1.1}$$

with $\mathcal{S}$ the subsampling operator, $\mathcal{F}$ the Fourier transform, y the subsampled measurements, J a regularisation functional that encourages the reconstruction to have a sparsity structure and α the regularisation parameter that controls the trade-off between the fit to measurements and fit to structure imposed by J. A prototypical example of a choice of regularisation functional is $J(u) = \|\mathcal{W}u\|_1$, where $\mathcal{W}$ is a wavelet transform and the ℓ^1-norm is used as a convex sparsity penalty.

Many previous efforts made towards accelerating MRI have focused on improving how these aspects are treated. The reconstruction algorithm can be changed to more accurately reflect the true structure of the signal: the typical convex reconstruction problem can be replaced by a dictionary learning approach (Ravishankar and Bresler, 2011b); in multi-contrast imaging, structural information obtained from one contrast can be used to inform a regularisation functional to use in the other contrasts (Ehrhardt and Betcke, 2016); and in dynamic MRI additional low rank structure can be exploited to improve reconstruction quality (Lingala et al., 2011; Trémoulhéac et al., 2014).

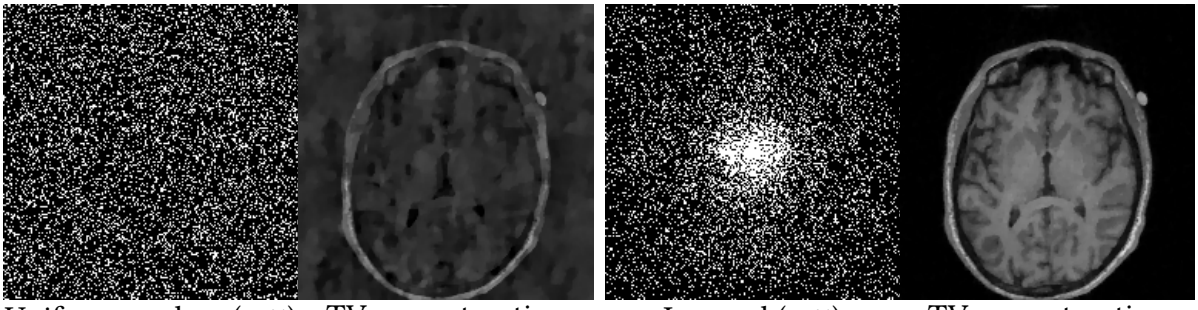

Figure 1.1: The importance of a good choice of sampling pattern. Left: uniform random pattern (sampling 19% of k-space) and reconstruction (using total variation type regularisation) on a test image. Right: an equally sparse pattern learned by our algorithm and reconstruction for the same test image.

It is well known that sampling uniformly at random in k-space (as the original compressed sensing theory suggests (Candès et al., 2006)) does not work well in practice; see Figure 1.1

for an empirical demonstration of this phenomenon and the discussion in Adcock et al. (2017) highlighting the fact that measurements are generally not incoherent. Using a variable density sampling pattern greatly improves reconstruction quality (Lustig et al., 2007a). Note that variable density sampling patterns of scattered points in k-space only allow for accelerated acquisition in 3D, in which case the readout is performed in the orthogonal direction. In the works Feng et al. (2014); Liu and Saloner (2014); Paquette et al. (2015); Piccini et al. (2011); Usman and Batchelor (2009), subsampling strategies are studied that can be used in practice. On the theoretical side, the compressed sensing assumptions have been refined to derive optimal densities for variable density sampling (Chauffert et al., 2014, 2013; Puy et al., 2011), to prove bounds on reconstruction errors for variable density sampling (Adcock et al., 2017; Krahmer and Ward, 2014) and to prove exact recovery results for Cartesian line sampling (Boyer et al., 2019; Poon, 2016).

The sampling pattern can be optimised in a given setting to improve reconstruction quality. There are works on fine-tuning sampling patterns (Ravishankar and Bresler, 2011a; Seeger et al., 2009), choosing data-adapted sampling patterns without knowledge of the reconstruction method (Knoll et al., 2011a), greedy algorithms to pick a suitable pattern for a given reconstruction method (Gözcü et al., 2018, 2019; Haldar and Kim, 2019), jointly learning a Cartesian line pattern and neural network reconstruction algorithm (Weiss et al., 2020), jointly learning non-Cartesian line sampling patterns and model based deep learning reconstruction algorithms for parallel MRI (Aggarwal and Jacob, 2020), and optimal patterns for zero-filling reconstructions can be computed from a training set with little computational effort (Li and Cevher, 2016). We consider the problem of learning an optimal sparse sampling pattern from scratch for a given variational reconstruction method and class of images by solving a bilevel optimisation problem. A similar approach has been used to learn regularisation parameters for variational regularisation models (De los Reyes et al., 2017), among other things. There has also been previous work in this direction, from the perspective of optimal experimental design, for ill-posed inverse problems with applications to geophysics and electrical impedance tomography (Horesh et al., 2010; Tenorio et al., 2013), but this work has been restricted to reconstructions using Tikhonov-type regularisations (corresponding to $J(u) = \|\mathcal{L}u\|^2$ for some linear operator $\mathcal{L}$).

1.1.1 Our contributions

In this chapter, we propose a novel bilevel learning approach to learn sparse sampling patterns for MRI. We do this within a supervised learning framework, using training sets of ground truth images with the corresponding measurements.

Our approach can accommodate arbitrary sampling patterns and sampling densities. We demonstrate that the parametrisation of the sampling pattern can be chosen to learn a pattern consisting of a scattered set of points as well as Cartesian lines, but other parametrisations can also be designed that result in radial or spiral sampling, for instance. By using a sparsity promoting penalty on the sampling pattern, we can also vary the sampling rates of our learned patterns.

Besides this, it is also possible to use a wide variety of variational reconstruction algorithms, that is various choices of regularisation J in Problem (1.1), and we can simultaneously learn the sampling pattern and the optimal regularisation parameter for reconstruction. This forgoes the need to separately tune the parameters of the reconstruction method.

Our optimal sampling patterns confirm empirically the validity of variable density sampling patterns: the optimal patterns tend to sample more densely around the low frequencies and more sparsely at high frequencies. We investigate the dependence of the shape of the sampling density on the sampling rate and the choice of regularisation functional R.

By focusing on a particular region within the body, our approach can be used with very small training sets to learn optimal patterns, that nevertheless generalise well to unseen MRI data. We demonstrate this on a set of brain images; indeed, in this setting we find that a training set of just five image, measurement pairs is sufficient.

1.2 Model and methods

In the bilevel learning framework, the free parameters of a variational regularisation method are learned to optimise a given measure of reconstruction quality. We assume that we are given a variational regularisation method to perform the reconstruction, of a form such as Problem (1.1). Furthermore, we assume that we are given a training set of N pairs of ground truth images u_i^* and fully sampled noisy k-space data y_i. With these ingredients we set up a bilevel optimisation problem that can be solved to learn the optimal sampling pattern $\mathcal{S}$ and regularisation parameter α:

$$\begin{aligned} &\min_{\mathcal{S},\alpha} \frac{1}{N} \sum_{i=1}^{N} L_{u_i^*}(\hat{u}_i(\mathcal{S},\alpha)) + P(\mathcal{S},\alpha) \\ &\text{where } \hat{u}_i(\mathcal{S},\alpha) \text{ solves Problem (1.1) with } y = y_i. \end{aligned} \tag{1.2}$$

In this problem, we use a continuous parametrisation of the sampling pattern (which is described in detail in Section 1.2.2) so that the learning problem is a continuous optimisation problem. A straightforward generalisation of this parametrisation (which is described in Section 1.A of the Appendix) allows us to impose constraints on the type of pattern that is learned. We will refer to Problem (1.2) as the upper level problem and will call the variational regularisation problems that make up its constraints the lower level problems. Each $L_{u_i^*}$ is a loss function that quantifies the discrepancy between the reconstruction from subsampled measurements, $\hat{u}_i$, and the corresponding ground truth u_i^* and P is a penalty on the sampling pattern that encourages its sparsity. Hence, the objective function in Problem (1.2) is a penalised empirical loss function, the minimiser of which trades off the reconstruction quality against the sparsity of the sampling pattern in an optimal manner. As we show in Section 1.2.3, it is possible to differentiate the solution maps $(\mathcal{S},\alpha) \mapsto \hat{u}_i(\mathcal{S},\alpha)$ in our setting, so that Problem (1.2) is amenable to treatment by first order optimisation methods.

In this section, we describe in more detail the various aspects that make up Problem (1.2) in our setting, starting with the lower level problems, followed by the upper level problem, after which we describe the methods that can be applied to solve the problem.

1.2.1 Variational regularisation models

The lower level problems in Problem (1.2) are variational regularisation problems. In this section, we specify the class of variational regularisation problems that will be considered. In our application, an image of dimensions $n := n_1 \times n_2$ is modelled as a vector in $\mathbb{C}^n$ by concatenating

its columns. The subsampled measurements corresponding to a given image u are modelled as $y = \mathcal{S}(\mathcal{F}u + \eta)$. Here $\mathcal{F}$ is a Fourier transform operator, $\mathcal{S} = \operatorname{diag}(s_1, \dots, s_n)$, $s_i \geqslant 0$ is the sampling operator, which selects the points in k-space that are included in the measurements (and can be used as a weight on those measurements), and $\eta \in \mathbf{C}^n$ is complex Gaussian white noise.

The variational regularisation approach to estimating the true image u from measurements y proceeds by solving an optimisation problem that balances fitting the measurements with fitting prior knowledge that is available about the image. In this chapter we consider problems that take the form of Problem (1.1) with $J(u) = Q(\mathcal{A}u)$. Here $\mathcal{A} : \mathbf{C}^n \to (\mathbf{C}^n)^M$ is a linear operator given by $\mathcal{A}u = (\mathcal{A}_1 u, \dots, \mathcal{A}_M u)$ for a collection of linear operators $\mathcal{A}_i : \mathbf{C}^n \to \mathbf{C}^n$, we let $|\mathcal{A}u|_i = \sqrt{|\mathcal{A}_1 u|_i^2 + \cdots + |\mathcal{A}_M u|_i^2}$, $\alpha \geqslant 0$, and $Q(v) = \sum_{i=1}^n \rho(|v|_i)$ for some convex $\rho : [0, \infty) \to \mathbf{R}$. Furthermore, we assume that ρ satisfies the following conditions:

- ρ is increasing,
- ρ is twice continuously differentiable,
- $\rho'(u) = O(u)$ as $u \to 0$.

Finally, a strongly convex penalty $u \mapsto \varepsilon \|u\|^2/2$ is added to the objective function. With these definitions, the lower level energy functional E_y, given fully sampled training measurements y, takes the following form:

$$E_y(u; \mathcal{S}, \alpha) = \frac{1}{2}\|\mathcal{S}(\mathcal{F}u - y)\|^2 + \alpha Q(\mathcal{A}u) + \frac{\varepsilon}{2}\|u\|^2 \tag{1.3}$$

Note that we can approximate a number of common regularisation functionals by choosing ρ to be defined as below for a small $\gamma > 0$:

$$\rho(x) = \begin{cases} -\frac{|x|^3}{3\gamma^2} + \frac{x^2}{\gamma} & \text{if} \quad |x| \leqslant \gamma \\ |x| - \frac{\gamma}{3} & \text{if} \quad |x| > \gamma. \end{cases}$$

This choice of ρ can be thought of as a twice continuously differentiable version of the Huber loss function (Huber, 1964). With this ρ, we obtain the following types of regularisation:

- if $\mathcal{A} = \nabla = (\partial_x, \partial_y)$ the regularisation term in Equation (1.3) approximates the isotropic total variation as regularisation term; its use in variational regularisation problems has been studied since Rudin et al. (1992), and it is a common choice of regularisation in compressed sensing MRI (Lustig et al., 2007a),

- if $\mathcal{A} = \mathcal{W}$ for some sparsifying transform $\mathcal{W}$, such as a wavelet or shearlet transform, the regularisation term in Equation (1.3) approximates a sparsity penalty on the transform coefficients of the image. These types of regularisation have been successfully applied to compressed sensing MRI in the past (Guerquin-Kern et al., 2009; Pejoski et al., 2015).

Hence, although this framework with smooth regularisation functionals precludes exactly using usual convex sparsifying transforms, we can approximate them closely.

1.2.2 The upper level problem

In the upper level problem, we parametrise the sampling pattern $\mathcal{S}$ and the lower level regularisation parameter α by a vector $p \in C := [0,1]^n \times [0,\infty)$: we let $s_i = p_i$ for $i = 1, \ldots, n$ and $\alpha(p) = p_{n+1}$. This parametrisation allows us to learn a sampling pattern of scattered points on a grid in k-space, though it is worth noting that the parametrisation can be generalised to constrain the learned pattern. To prevent the notation from becoming overly cumbersome, we do not consider this generalisation here, but refer the reader to Section 1.A in the Appendix for the details.

With this parametrisation, a natural choice of the sparsity penalty P is

$$P(p) = \beta \sum_{i=1}^{n} (p_i + p_i(1 - p_i))$$

with $\beta > 0$ a parameter that decides how reconstruction quality is traded off against sparsity of the sampling pattern. Besides encouraging a sparse sampling pattern, this penalty encourages the weights in the sampling pattern $\mathcal{S}(p)$ to take either the value 0 or 1. For the loss function L, we choose $L_{u'}(u) = \frac{1}{2}\|u - u'\|^2$, but it is straightforward to replace this by any other smooth loss function. For instance, if one is interested in optimising the quality of the recovered edges one could use the smoothed total variation as a loss function: $L_{u'}(u) = \sum_{i=1}^{n} h_\gamma(|\nabla u' - \nabla u|_i)$, with h_γ as defined in Section 1.2.1.

1.2.3 Methods

As was mentioned in Section 1.2, first order optimisation methods can be used to solve problems like Problem (1.2), provided that the solution maps of the lower level problems, $p \mapsto \hat{u}_i(p)$, can be computed and can be differentiated. In this section we describe the approach taken

to computing the solution maps and their derivatives and then describe how these steps are combined to apply first order optimisation methods to Problem (1.2).

Computing the solution maps of the lower level problems

In this and the next subsection, we will consider the lower level problem for a fixed y, so for the sake of notational clarity, we will drop the subscript and write $E = E_y$. The lower level energy functional E is convex in u and takes the saddle-point structure that is used in the primal-dual hybrid gradient algorithm (PDHG) of Chambolle and Pock (Chambolle and Pock, 2011). Indeed, we can write

$$E(u; \mathcal{S}(p), \alpha(p)) = F(\mathcal{K}u) + G(u),$$

with $F : \mathbf{C}^d \times (\mathbf{C}^d)^M \to \mathbf{R}$, $(v_1, v_2) \mapsto F_1(v_1) + F_2(v_2)$, $\mathcal{K} : \mathbf{C}^d \to \mathbf{C}^d \times (\mathbf{C}^d)^M$, $u \mapsto (u, \mathcal{A}u)$ and $G : \mathbf{C}^d \to \mathbf{R}$, where

$$\begin{aligned} F_1(v_1) &= \frac{1}{2}\|\mathcal{S}(p)(\mathcal{F}v_1 - y)\|^2, \\ F_2(v_2) &= \alpha(p)Q(v_2), \\ G(u) &= \frac{\varepsilon}{2}\|u\|^2. \end{aligned}$$

Since the lower level problems are strongly convex and smooth, it is possible to obtain linear convergence rates when applying first-order methods such as PDHG to solve them. The splitting given above and the parameter choices from Section 1.C.2 in the appendix, combined with an arbitrary initialisation u^0 (we can take it to be the zero-filling reconstruction, or warm start the solver) ensure that PDHG attains this linear convergence rate. The details of applying PDHG are described in Algorithm 1:

The lower level problems are smooth, so a natural alternative to PDHG is to consider using a second-order solver to obtain (at least locally) superlinear convergence rates. Since E is also strongly convex, its Hessian is positive definite, implying that an approximate step can be computed in Newton's method by running an inner loop of the conjugate gradient (CG) method. This requires us to compute Hessian-vector products, which are also required for Section 1.2.3, and for which we have explicit expressions (see Section 1.D in the appendix). Combining this with a line search that ensures the objective function value decreases sufficiently, we obtain a Newton-CG algorithm (Nocedal and Wright, 2006) as shown in Algorithm 2:

Let us compare the performance of the Newton-CG solver to the PDHG solver for some example lower level problems. Since the Newton-CG solver has an inner loop, each iteration

Algorithm 1 Solving the lower level problem, Problem (1.1), with PDHG

inputs: u^0, `maxit`, `tol`
$v^0 \leftarrow \mathcal{K}u^0$
$\overline{u}^0 \leftarrow u^0$
for $k = 0$ to `maxit` **do**
  $v^{k+1} \leftarrow \text{prox}_{\sigma F^*}(v^k + \mathcal{K}\overline{u}^k)$
  $u^{k+1} \leftarrow \text{prox}_{\tau G}(u^k - \tau\mathcal{K}^* v^{k+1})$
  $\overline{u}^{k+1} = u^{k+1} + \theta(u^{k+1} - u^k)$
  if $\frac{\|u^{k+1}-u^k\|}{\|u^k\|} + \frac{\|v^{k+1}-v^k\|}{\|v^k\|} \leqslant$ `tol` **then**
    break the loop
  end if
end for
return u^{k+1}

Algorithm 2 Solving the lower level problem, Problem (1.1), with Newton-CG using the Armijo line search

inputs: u^0, `maxit`, `maxls`, τ, c
for $k = 0$ to `maxit` **do**
  Use CG to solve the equation $D_u^2 E(u^k; p)d = D_u E(u^k; p)$ for d to obtain d^{k+1}
  $m^{k+1} \leftarrow \langle d^{k+1}, D_u E(u^k; p)\rangle$
  $\sigma \leftarrow 1; i \leftarrow 0$
  while $E(u^k - \sigma d^k; p) - E(u^k; p) > -c\sigma m^{k+1}$ and $i <$ `maxls` **do**
    $\sigma \leftarrow \tau\sigma; i \leftarrow i + 1$
  end while
  if $i =$ `maxls` **then**
    break the loop
  end if
  $u^{k+1} \leftarrow u^k - \sigma d^k$
end for
return u^{k+1}

of which is comparable in cost to an iteration of the PDHG solver, we should count its inner iterations for fairness of the comparison. We initialise both solvers using $u^0 = 0$ and set the parameters of Newton-CG to be $\tau = 0.5, c = 10^{-3}$, `maxls` $= 20$. Figure 1.2 shows the result of comparing the two solvers on two representative lower level problems with TV-type regularisation. We measure progress in terms of the gradient norm $\|D_u E(u; p)\|$. Evidently, PDHG makes steady progress before stagnating, whereas Newton-CG initially makes slow progress, before entering the superlinear convergence regime and attaining a final result that is closer to optimal than the final result of PDHG. Note, however, that in the case of the sparse sampling pattern it takes over 2000 iterations before Newton-CG enters the superlinear

convergence regime. Although Newton-CG attains a more accurate solution than PDHG, we will see in the next section that the PDHG solutions are sufficiently accurate for use in differentiating the solution map. Since the PDHG solutions are cheaper to compute, we will use them in the inner loop of the overall bilevel optimisation problem.

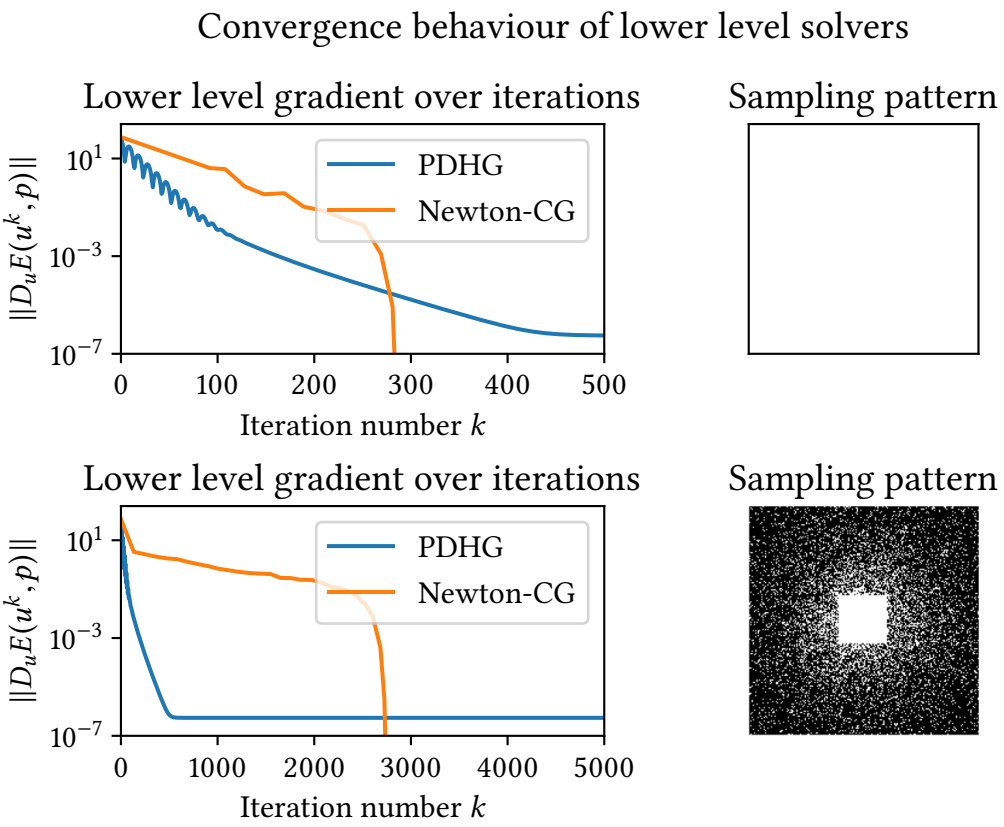

Figure 1.2: A comparison of the PDHG and Newton-CG solvers for the lower level problem. We use a TV-type regularisation, with regularisation parameter $\alpha = 10^{-2}$ and the sampling patterns displayed on the right. The Newton-CG eventually reaches a superlinear convergence regime, but for the sampling pattern which is sparse (as the majority of the inner iterates of the bilevel optimisation problem will be), it takes excessively long before this happens. Note in particular the range of the horizontal axis on the bottom plot. On the other hand, PDHG makes steady progress towards an approximate solution that is adequate.

Differentiating the solution map

In the previous subsection, we saw that we can compute the solution maps of the lower level problems. To apply first order optimisation methods to Problem (1.2), we still need to be able to differentiate these solution maps. To this end, note that the solution map $\hat{u}$ of E can be

defined equivalently by its first order optimality condition:

$$D_u E(\hat{u}(p); p) = 0$$

and that E is twice continuously differentiable in our setting. To ease notation, let us write $\hat{u}_p := \hat{u}(p)$ in this subsection. Since E is strongly convex in u, its Hessian is positive definite and hence invertible. As a consequence, the implicit function theorem tells us that the optimality condition can be implicitly differentiated with respect to p and solved to give the derivative of the solution map:

$$D_u^2 E(\hat{u}_p; p) D_p \hat{u}_p + D_{u,p} E(\hat{u}_p; p) = 0,$$

so that

$$D_p \hat{u}_p = -[D_u^2 E(\hat{u}_p; p)]^{-1} D_{u,p} E(\hat{u}_p; p). \tag{1.4}$$

In fact, we do not need the full derivative of the solution map in our application, but just the gradient of a scalar function of the solution map, namely $p \mapsto L_{u^*}(\hat{u}_p)$ for some ground truth u^*. The chain rule and the formula in Equation (1.4) give us a formula for this gradient:

$$\begin{aligned} g &= \nabla_{\hat{u}_p} L_{u^*}(\hat{u}_p) D_p \hat{u}_p \\ &= -\nabla_{\hat{u}_p} L_{u^*}(\hat{u}_p) [D_u^2 E(\hat{u}_p; p)]^{-1} D_{u,p} E(\hat{u}_p; p) \\ &= -D_{p,u} E(\hat{u}_p; p) [D_u^2 E(\hat{u}_p; p)]^{-1} \nabla_{\hat{u}_p} L_{u^*}(\hat{u}_p)^*. \end{aligned} \tag{1.5}$$

It is worth noting that this expression for the gradient can also be derived using the Lagrangian formulation of Problem (1.2), through the adjoint equation, and this is the way in which it is usually derived when an optimal control perspective is taken (De los Reyes et al., 2017). To implement this formula in practice, we do not compute the Hessian matrix of E and invert it exactly (since the Hessian is very large; it has as many rows and columns as the images we are dealing with have pixels). Instead, we emphasise that the Hessian is symmetric positive definite, so that it is suitable to solve the linear system with an iterative solver such as the conjugate gradient method. For this, we just need to compute the action of the Hessian, for which we can give explicit expressions. These computations have been done in Section 1.D of the appendix. The expressions derived in the appendix for $D_u^2 E$ and $D_{p,u} E$ can be implemented efficiently in practice and are then used in the conjugate gradient method (CG) to compute the desired gradients.

As mentioned in the previous section, we use PDHG to approximately solve the lower level problems. In particular, this means in practice that we are not solving Equation (1.5), but $\hat{u}_p$ on the right-hand side is replaced by an iterate u^k of PDHG applied to the relevant lower

level problem. Let us study how the corresponding approximate gradient g^k, computed by solving (using CG with a tiny tolerance) the equation

$$g^k = -D_{p,u}E(u^k;p)[D_u^2E(u^k;p)]^{-1}\nabla_{\hat{u}}L_{u^*}(u^k)^*, \tag{1.6}$$

converges to the true gradient g of Equation (1.5). Since we can not explicitly solve the lower level problem and adjoint equation in our setting, we need a reasonable estimate of $\hat{u}_p$ and g. As we noted in the previous section, Newton-CG may initially be slow at solving the lower level problems, but after entering the superlinear convergence regime, the found solution is highly accurate. Hence, we will use the output of Newton-CG to estimate $\hat{u}_p$, and will use CG (again with a tiny tolerance) to solve Equation 1.5 to estimate g. The results of doing this in the same settings that were initially studied in Figure 1.2 are displayed in Figure 1.3. We see that the approximate gradients g^k converge to g in much the same way as the approximate solutions u^k converge to $\hat{u}_p$, until they stagnate at a relative error around 10^{-6}. The upper level solver described in the next section uses a line search procedure, which can fail when the upper level gradients are too inaccurate. The gradient errors attained in Figure 1.3 after a few hundred iterations of PDHG are sufficiently small to ensure that no such problems occur.

Solving the bilevel problem using L-BFGS-B

Recall that we are interested in solving Problem (1.2). By the previous sections, we know that the objective function of this problem is continuously differentiable, and the constraints that we impose on the parameters form a box constraint, so the optimisation problem that we consider is amenable to treatment by the L-BFGS-B algorithm (Byrd et al., 1995; Zhu et al., 1997); L-BFGS-B is a limited-memory (explaining the L in the acronym) version of the quasi-Newton optimisation method of Broyden, Fletcher, Goldfarb and Shanno (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1971) that can handle box constraints (explaining the final B in the acronym). In our description of the computation of the objective function value and gradient of Problem (1.2), we will denote the gradient of $p \mapsto L_{u_i^*}(\hat{u}_i(p))$ by g_i. Since the objective function splits as a sum over the training set, it is completely straightforward to parallelise the computations of the solution maps and desired gradients over the training set in Algorithm 3. The output of algorithm 3 can be plugged in to L-BFGS-B to solve Problem (1.2).

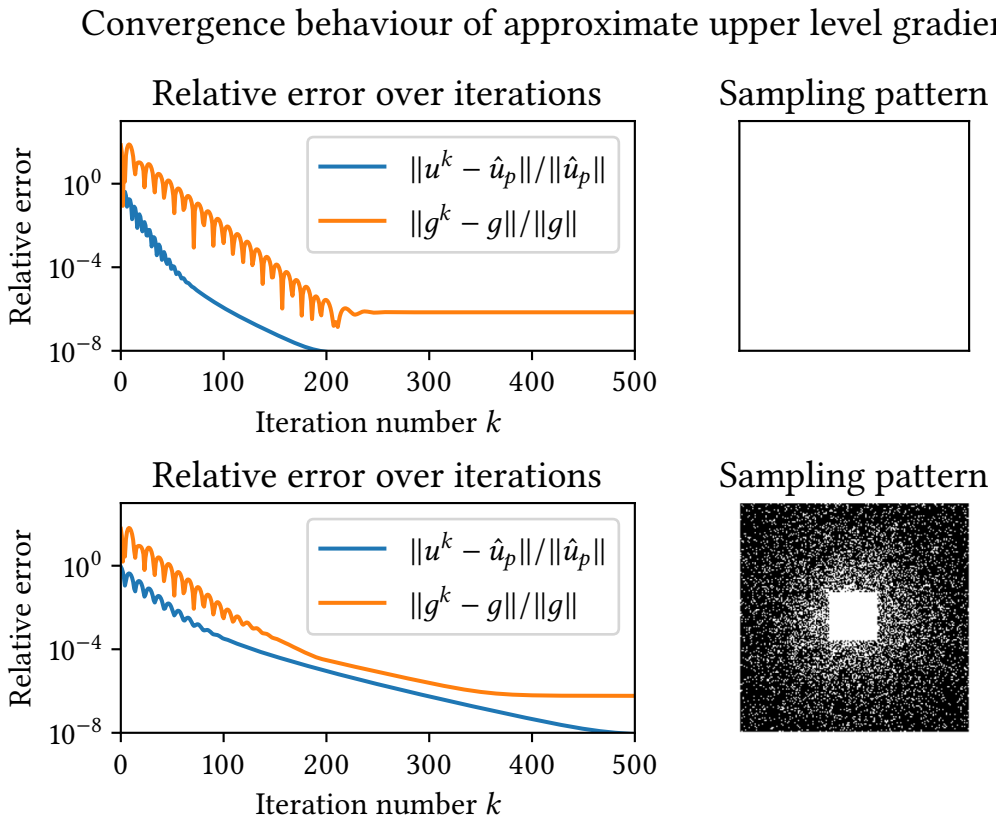

Figure 1.3: Convergence of the upper level gradient, computed using the implicit differentiation approach of Equations (1.5) and (1.6), as a function of the number of iterations of the lower level solver PDHG. We consider the same settings as shown in Figure 1.2: we use a TV-type regularisation with $\alpha = 10^{-2}$ and the sampling patterns shown on the right.

Algorithm 3 Computing the objective function value L and gradient g of the bilevel problem, Problem (1.2), at p

input: p
for $i = 1$ to N **do**
  Set measurements for training example i: $y \leftarrow y_i$
  Set current $\mathcal{S}$ and α: $\mathcal{S} \leftarrow \mathcal{S}(p), \alpha \leftarrow \alpha(p)$
  Solve Problem (1.1) with Algorithm 1 to obtain $\hat{u}_i$
  Solve the system in Equation (1.5) with CG to obtain g_i
end for
$L \leftarrow \frac{1}{N}\sum_{i=1}^{N} L_{u_i^*}(\hat{u}_i) + P(p)$
$g \leftarrow \frac{1}{N}\sum_{i=1}^{N} g_i + \nabla_p P(p)$
return L, g

1.3 Experiments

Our methods have been implemented in Python, using the PyTorch package (Paszke et al., 2019) to solve the lower level problems and adjoint equations (Equation (1.5)). We implement the lower level solver as a custom PyTorch module with the backpropagation given by solving the adjoint equation, which allows it to be easily used as a component in another machine learning problem and enables us to make use of GPUs to accelerate computations if available. Our code is available at `https://github.com/fsherry/bilevelmri`. We use the implementation of the L-BFGS-B algorithm that is included in SciPy (Virtanen et al., 2020) and a PyTorch implementation of the discrete wavelet transform (Cotter and McLaughlin, 2019) for our experiments involving wavelet regularisation. All experiments were run on a computer with an Intel Xeon Gold 6140 CPU and a NVIDIA Tesla P100 GPU. Since the learning problem is nonconvex, care must be taken with the choice of initialisation. In the experiments in this section, we initialise the learning with a full sampling pattern and the corresponding optimal regularisation parameter. This optimal regularisation parameter is learned using our method, keeping the sampling pattern fixed to fully sample k-space; the optimal regularisation parameter is typically found in less than 10 iterations of the L-BFGS-B algorithm. In practice, this initialisation is found to work well.

In this section, we have experiments in which we look at

- varying the sparsity parameter β to control the sparsity of the learned pattern,
- learning Cartesian line patterns with our method,
- using different lower level regularisations,
- varying the size of the training set,
- comparing the learned patterns to other sampling patterns,
- learning sampling patterns for high resolution imaging.

Unless otherwise specified, we use a total variation type regularisation in the lower level problems for all experiments. That is, ρ is chosen as the Huber type function defined in Section 1.2.1 and $\mathcal{A} = \nabla$. We refer the reader to the supporting document for figures that may be of interest, but are not crucial to the understanding of the results.

1.3.1 Data

The brain images are of size 192×192, taken as slices from 7 separate T1-weighted 3D scans. The corresponding noisy measurements are simulated by taking discrete Fourier transforms of these slices and adding complex Gaussian white noise. In all experiments except the one in Section 1.3.5, we use a training set consisting of 7 slices. We use 70 slices different to those used in training to test the performance of learned patterns. The scans were acquired on a Siemens PrismaFit scanner. For all scans except one, TE = 2.97 ms, TR = 2300 ms and the Inversion Time was 1100 ms. For the other scan, TE = 2.98 ms, TR = 2300 ms and the Inversion Time was 900 ms.

The brain images used in the experiments shown in Figure 1.12 are of size 217×181, taken as slices from a simulated T2-weighted 3D scan from the BrainWeb database (Cocosco et al., 1997). Noisy measurements are simulated from these slices by taking discrete Fourier transforms and adding complex Gaussian white noise. We use a training set consisting of 5 slices and we use 5 slices different to those used in training to test the performance of learned patterns. In these experiments, the corresponding slices from the T1-weighted scan are used to inform the directional vector fields that are used in the directional total variation regularisation (Ehrhardt and Betcke, 2016) in the lower level problems.

The high resolution images are of size 1024×1024, taken as slices from a T1-weighted 3D scan of a test phantom. We use a training set consisting of 5 slices and test the learned pattern on a single slice different to the ones used in training. Again, the noisy measurements are simulated by taking discrete Fourier transforms of these slices and adding complex Gaussian white noise. The scan was acquired on a GE 3T scanner using spoiled gradient recalled acquisition with TE = 12 ms and TR = 37 ms.

1.3.2 Varying the sparsity parameter β

Learning with a training set of 7 brain images, we consider the effect of varying the sparsity parameter β. Increasing this parameter tends to make the learned patterns sparser, although we do see a slight deviation from this monotone behaviour for large β. Figure 1.4 shows examples of the learned patterns and reconstructions on a test image and in Figure 1.5, we see the performance of the learned patterns, evaluated on the test set of 70 brain images. We use a Gaussian kernel density estimator to estimate a sampling distribution corresponding to each pattern. That is, we convolve the learned pattern with a Gaussian filter with a small bandwidth and normalise the resulting image to sum to 1. The results of doing this can be seen in Figure 1.6: we see that the distributions become more peaked strongly around the origin as

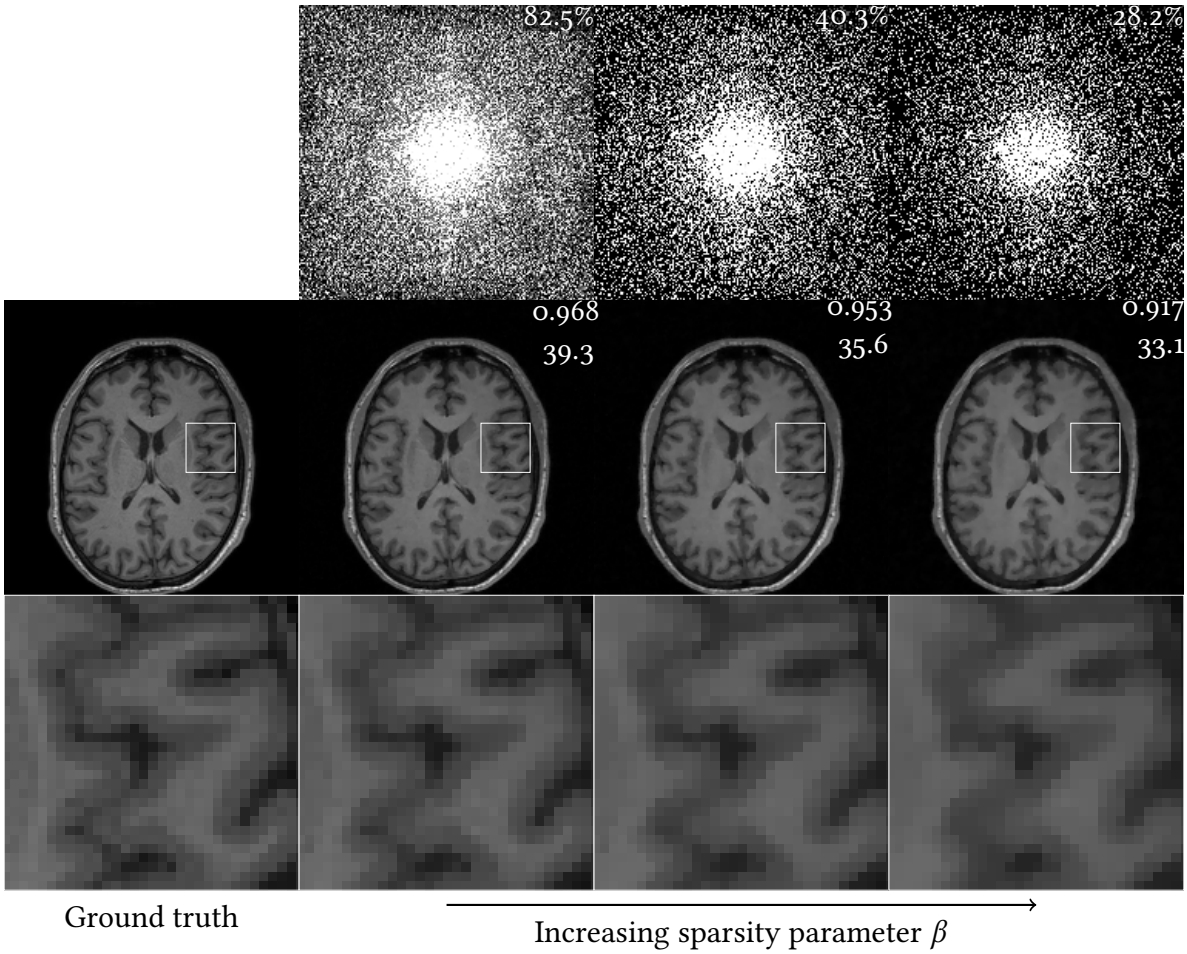

Figure 1.4: Learned sampling patterns and the corresponding reconstructions on a test image with TV regularisation in the lower level problem. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR. The values of β used were (from left to right) $1.58 \cdot 10^{-4}, 1.58 \cdot 10^{-3}, 1.58 \cdot 10^{-2}$.

the patterns become sparser and furthermore, we see that the decay in the learned patterns is anisotropic (as opposed to the isotropic decay of variable density sampling patterns that are not adapted to the data, such as in Lustig et al. (2007a)).

1.3.3 Cartesian line sampling

As described in Section 1.A of the Appendix, we can restrict the learned pattern to sample along Cartesian lines. Similarly to the case of learning scattered points in k-space, we see in Figure 1.7 that we have some control over the sparsity of the learned pattern using the parameter β. The sparsity penalty P does not seem to work as well in this situation in encouraging the weights of the pattern to be binary, so we threshold the resulting patterns (that is, we take

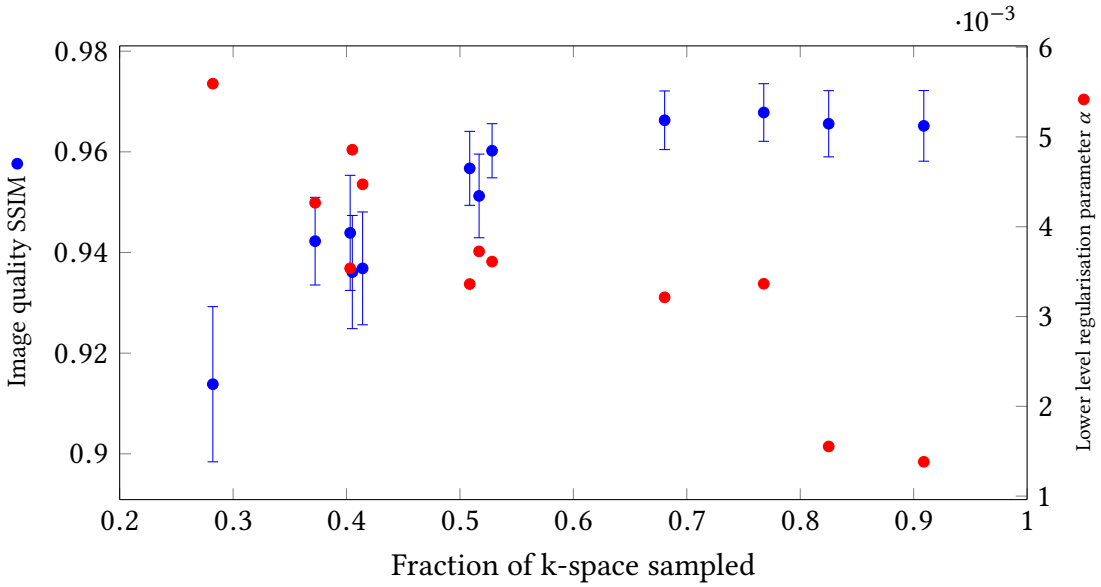

Figure 1.5: Performance of the learned patterns (measured using the SSIM index) on the test set, and the lower level regularisation parameter α that was learned, against the fraction of k-space that is sampled.

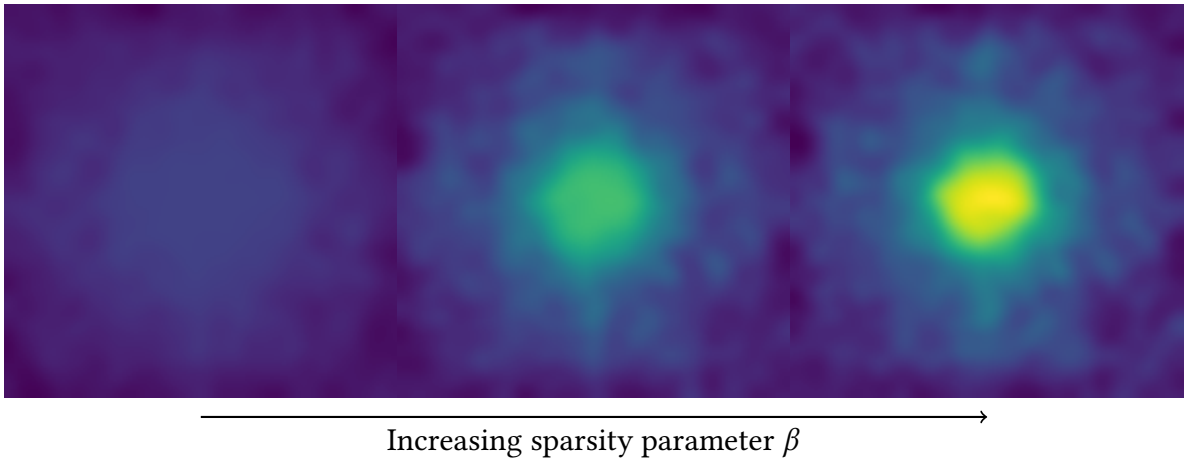

Figure 1.6: Gaussian kernel density estimates of the sampling distributions for reconstruction with TV regularisation.

$p_i^{\text{thresholded}} = 1$ if $p_i > 0$ and $p_i^{\text{thresholded}} = 0$ if $p_i = 0$) and tune the lower level regularisation parameter on the training set using our method and the thresholded pattern.

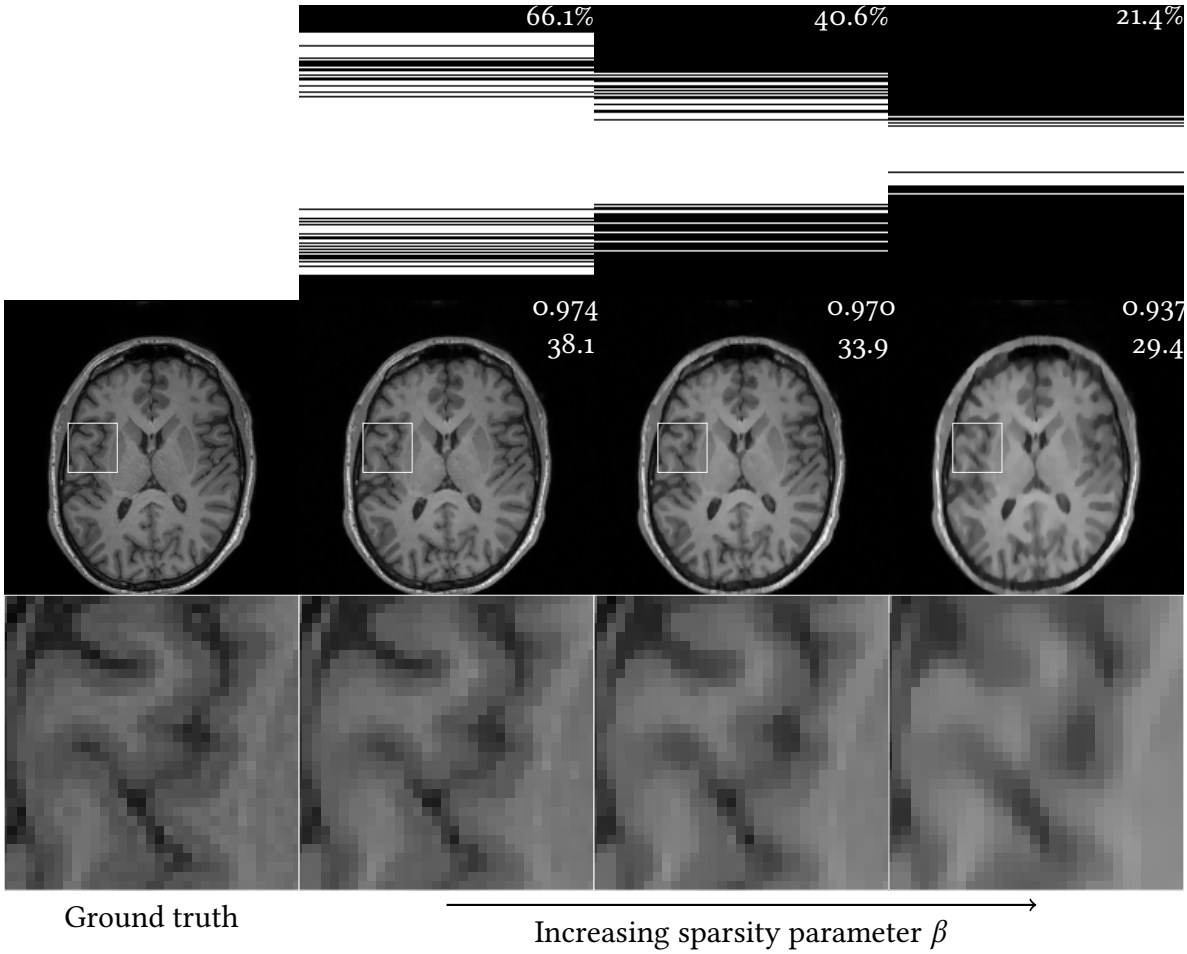

Figure 1.7: Learned Cartesian line sampling patterns and the corresponding reconstructions on a test image with TV regularisation in the lower level problem. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR. The values of β used were (from left to right) $1.58 \cdot 10^{-3}, 6.31 \cdot 10^{-3}, 1.58 \cdot 10^{-2}$.

1.3.4 Other lower level regularisations

Wavelet regularisation

Instead of the TV type regularisation, we use a sparsity penalty on the wavelet coefficients of the image. We accomplish this by choosing $\rho = h_\gamma$ and $\mathcal{A} = \mathcal{W}$ for $\mathcal{W}$ an orthogonal wavelet transform (we use Daubechies-4 wavelets). This results in learned sampling patterns that have slightly different qualititative properties compared to those for the total variation regularisation. Comparing two patterns from the TV and wavelet regularisation with the same sparsity, we find that the pattern for the wavelet regularisation is more strongly peaked around the origin. We can see this in Figure 1.8, where we have estimated the sampling

distributions for two learned patterns with TV and wavelet regularisation, both of which sample approximately 27% of k-space.

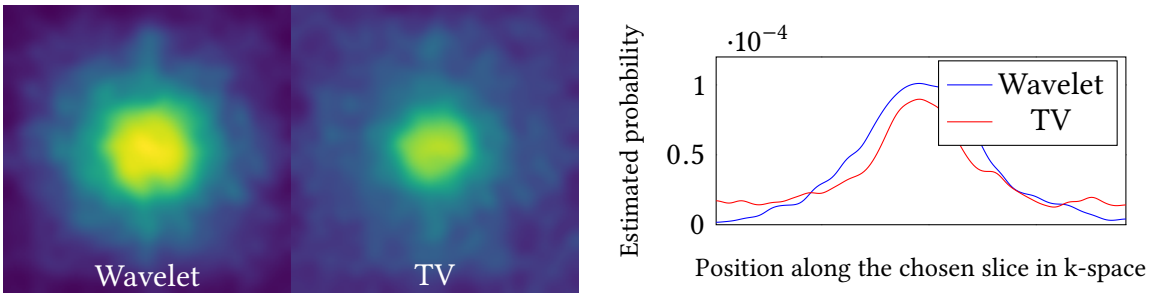

Figure 1.8: Gaussian kernel density estimates of the sampling distributions for reconstruction with wavelet and TV regularisation (for approximately the same sparsity in k-space). On the right we plot slices taken along the diagonal of these distributions, showing clearly that the sampling distribution for reconstruction with wavelet regularisation is more strongly peaked around the centre.

H^1 regularisation

We use the squared H^1 seminorm as lower level regularisation, if we take $\rho(x) = x^2/2$ and $\mathcal{A} = \nabla$ in the lower level problem. With this choice, we find that the learned α equals 0 and that the learned pattern does not take on just binary values: the weights of the learned pattern are lower at higher frequencies, as can be seen in Figure 1.9.

No regularisation

Taking no regularisation in the lower level problem, i.e. $\rho = 0$ and fixing $\alpha = 0$, we find essentially the same results as when we considered the H^1 regularisation: the weights in the learned pattern show a decay away from the origin as in Figure 1.9.

Comparison of the different regularisations

We compare the performance of the learned patterns with the different lower level regularisations. In Table 1.1, we list the performance of three of these patterns on the test set of brain images, each pattern sampling roughly the same proportion of k-space.

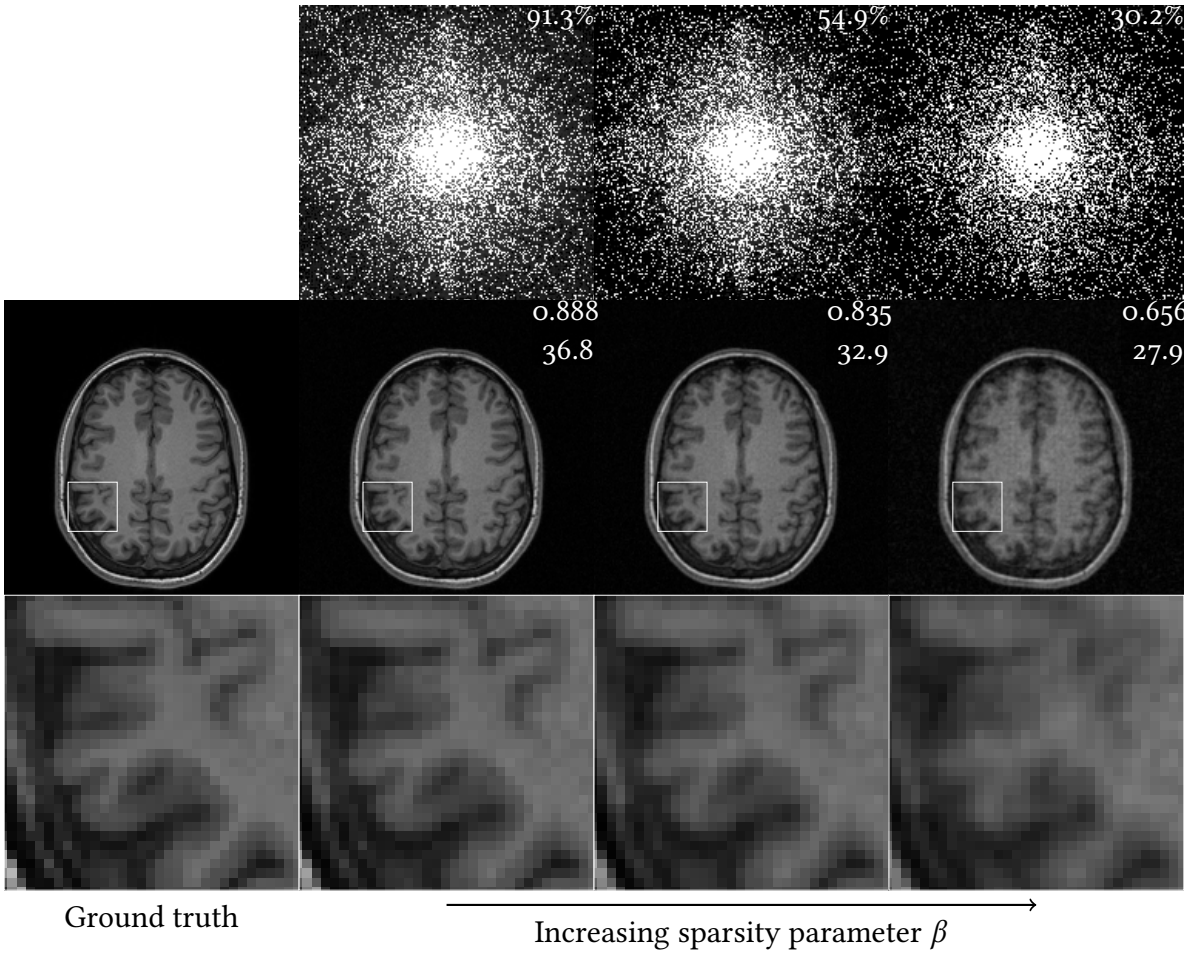

Figure 1.9: Learned sampling patterns and the corresponding reconstructions on a test image with H^1 regularisation in the lower level problem. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR. The values of β used were (from left to right) 10^{-3}, $2.51 \cdot 10^{-3}$, $6.31 \cdot 10^{-3}$.

Table 1.1: Performance of the learned patterns with different lower level regularisation functionals.

	Regularisation	SSIM	PSNR
Training	TV (28.2%)	0.980 ± 0.002	31.6 ± 0.5
	Wavelet (25.7%)	0.962 ± 0.003	29.3 ± 0.4
	H^1 (30.2%)	0.872 ± 0.004	25.9 ± 0.3
Testing	TV (28.2%)	0.915 ± 0.002	33.1 ± 0.7
	Wavelet (25.7%)	0.913 ± 0.001	31.9 ± 0.7
	H^1 (30.2%)	0.651 ± 0.005	28.1 ± 0.5

The TV regularisation is seen to outperform wavelet regularisation, which in turn outperforms H^1 regularisation. Figure 1.10 shows the three patterns that we are comparing and the corresponding reconstructions on a test image. We note that this method can easily be extended to other regularisation functions (such as the Total Generalised Variation) that have been used in the context of MRI (Benning et al., 2014; Knoll et al., 2011a).

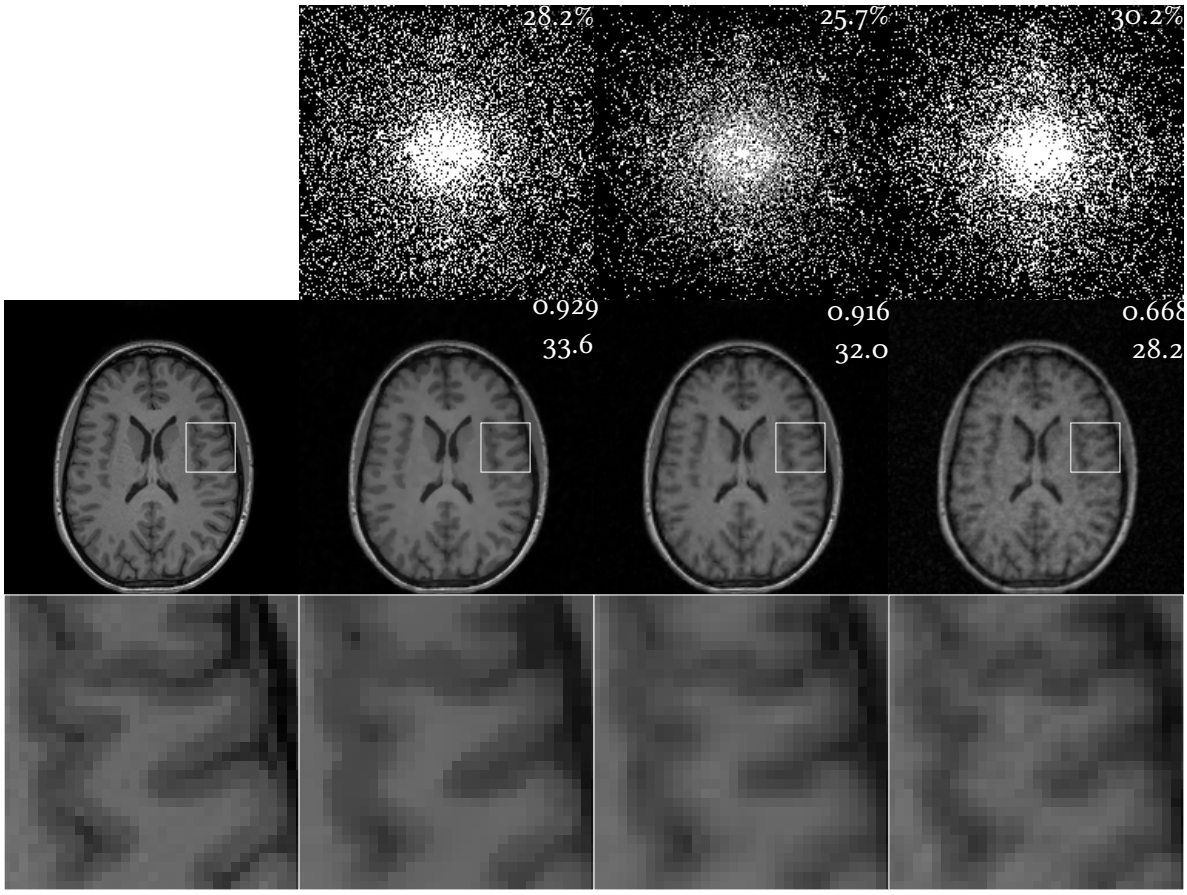

Figure 1.10: A comparison of learned sampling patterns for the different lower level regularisations that we have considered. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR.

1.3.5 Varying the size of the training set

To investigate the effect of the size of the training set, we ran our method on different training sets of slices of brain images, of sizes 1, 3, 5, 10, 20, 30 to obtain sampling patterns of roughly the same sparsity. As we see in Figure 1.11, the learned patterns perform reasonably well (on

the training set of 70 slices) from a training set of size 5 and performance flattens out as the size of the training set increases to about 20.

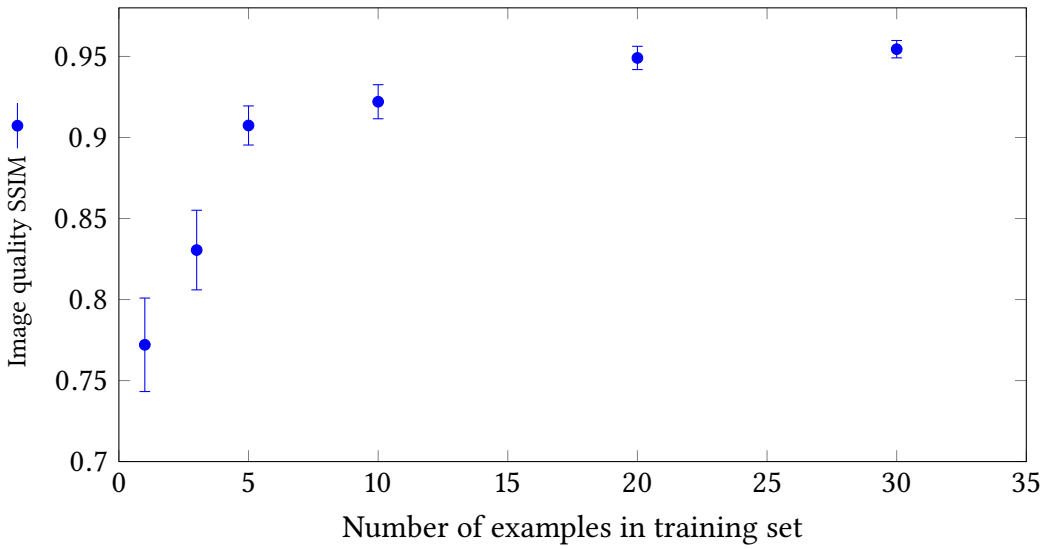

Figure 1.11: The performance of the learned pattern on the test set as it depends on the size of the training set.

1.3.6 Comparing with other patterns

In this subsection, we compare the performance of our learned patterns to the performance of sampling patterns chosen using other strategies. Section 1.3.6 considers the problem of choosing a sampling pattern of scattered 2D points, while Section 1.3.6 discusses the case where sampling is constrained to Cartesian lines.

Free patterns

We compare our method for learning sampling patterns to a different data-adapted method for generating sampling patterns (Knoll et al., 2011b) and to uninformed variable density sampling patterns as in Lustig et al. (2007a). In this comparison, we use directional total variation regularisation (Ehrhardt and Betcke, 2016) in the lower level problem. We use slices from a T1-weighted 3D scan from the BrainWeb database (Cocosco et al., 1997) to generate reference vector fields and use the corresponding slices from the T2-weighted scan as ground truths. The pattern is learned with a training set of 5 slices and checked on a testing set of 5 slices. Neither the data-adapted pattern from Knoll et al. (2011b) nor the uninformed variable density sampling pattern from Lustig et al. (2007a) fix the lower level regularisation parameter, so we

fix these by using our method to learn the optimal regularisation parameter on the training set. The directional total variation is a strong form of regularisation since edge information from one modality is used to regularise the reconstruction of another modality. As a result, we see in Figure 1.12 and Table 1.2 that reconstructions with all of the patterns are relatively good, even at a low sampling rate. Comparing the details we see that both of the data-adapted patterns outperform the uninformed variable density sampling pattern, and that our learned pattern outperforms both other patterns. Since our pattern was learned using knowledge of the lower level regularisation and the pattern from Knoll et al. (2011b) does not use this information, we conclude that it is possible to adapt to the reconstruction method to improve sampling strategies. The zoomed regions in Figure 1.12 show that our method does a better job at resolving the fine structures in the image.

Table 1.2: A comparison of the performance of our learned pattern to the data-adapted patterns of Knoll et al. (2011b) and uninformed variable density sampling patterns from Lustig et al. (2007a) with dTV regularisation in the lower level problem. All compared sampling patterns sample 13.2% of k-space.

	Pattern type	SSIM	PSNR
Training	Our method	0.977 ± 0.002	32.5 ± 0.2
	Data-adapted (Knoll et al., 2011b)	0.968 ± 0.002	31.1 ± 0.1
	Uninformed VDS (Lustig et al., 2007a)	0.925 ± 0.005	28.9 ± 0.1
Testing	Our method	0.975 ± 0.003	32.1 ± 0.2
	Data-adapted (Knoll et al., 2011b)	0.967 ± 0.003	31.1 ± 0.2
	Uninformed VDS (Lustig et al., 2007a)	0.924 ± 0.003	28.8 ± 0.1

Cartesian line patterns

Finally, we compare our method for Cartesian line patterns to another recent method for learning sampling patterns (Gözcü et al., 2018) and uninformed variable density sampling patterns (Lustig et al., 2007a). In the method of Gözcü et al. (2018), a set of candidate masks is considered and a sampling pattern is selected by adding candidate masks one at a time according to a greedy selection rule: at each stage, the candidate is chosen among the remaining candidates that gives the maximum increase of a performance measure on a training set. A drawback of the method from Gözcü et al. (2018) is that the lower level regularisation parameter, has to be fixed beforehand; we fix the regularisation parameter learned with

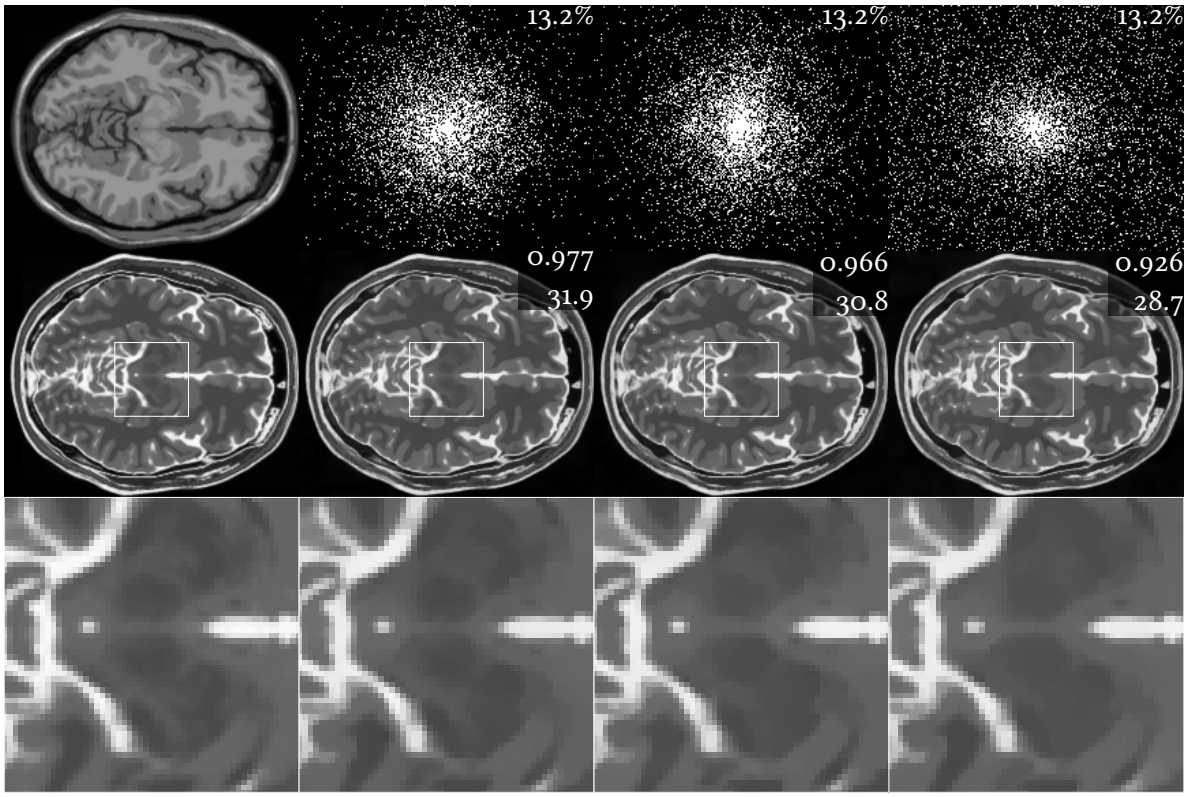

Figure 1.12: A comparison of our learned pattern to another data-adapted pattern (Knoll et al., 2011b) and an uninformed variable density sampling pattern (Lustig et al., 2007a) with dTV regularisation in the lower level problem. The example image shown is a test example, not seen by our learned method or the data-adapted method at training time. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR. The top image in the ground truth column is the T1-weighted slice that is used to generate the reference vector field for the dTV regularisation for this test example.

our method on the training set, apply the method from Gözcü et al. (2018) to learn a line pattern, and finally tune the regularisation parameter on the training set with our method to improve the performance of the pattern learned with the method from Gözcü et al. (2018). The uninformed variable density sampling pattern from Lustig et al. (2007a) does not fix the reconstruction method, so we use our method to learn the optimal regularisation parameter on the training set for this sampling pattern. We use a training set of 7 slices and test on 70 slices different to the ones used in training.

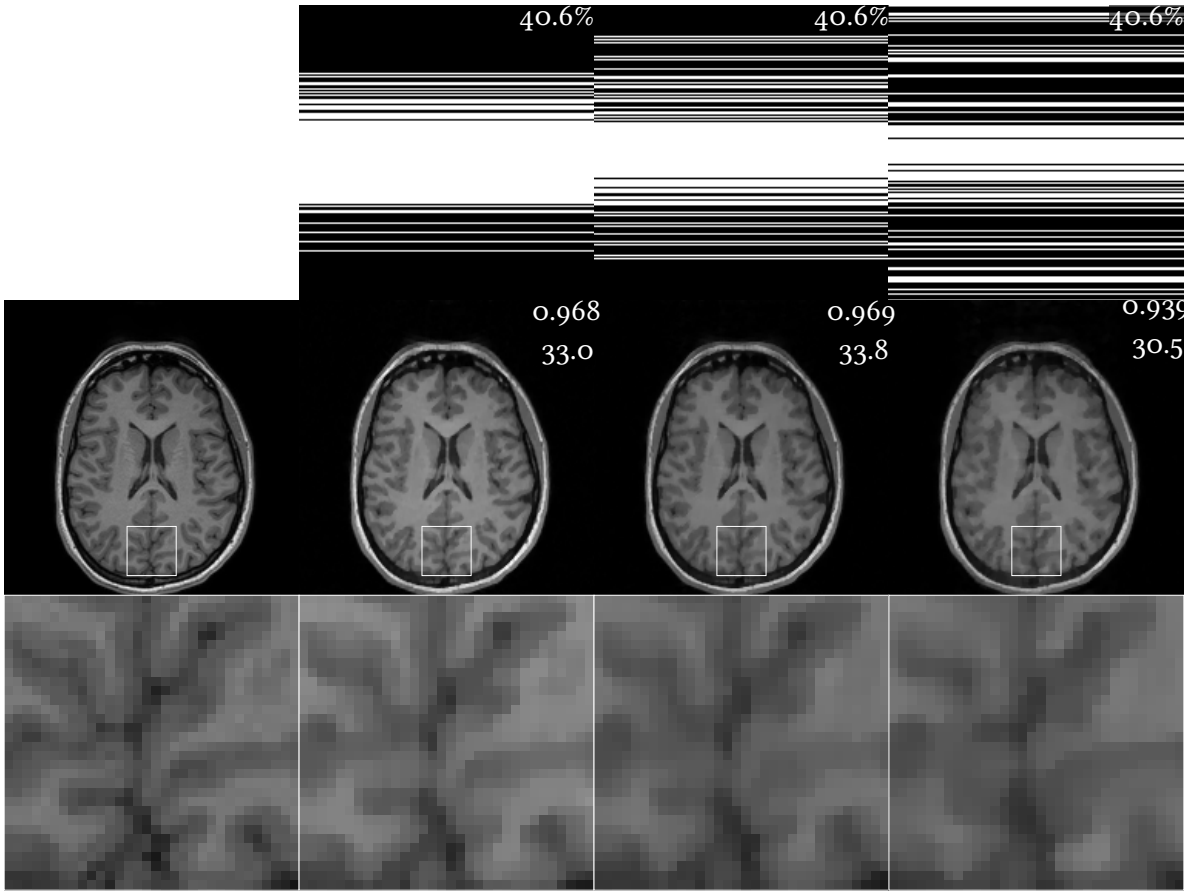

Figure 1.13: A comparison of our learned Cartesian line pattern to the learned pattern from Gözcü et al. (2018) and an uninformed variable density sampling pattern (Lustig et al., 2007a) with TV regularisation in the lower level problem. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR.

Table 1.3: A comparison of the performance of our learned Cartesian line pattern to the learned patterns of Gözcü et al. (2018) and uninformed variable density sampling patterns from Lustig et al. (2007a) with TV regularisation in the lower level problem. All compared sampling patterns sample 40.6% of k-space.

	Pattern type	SSIM	PSNR
Training	Our method	0.978 ± 0.002	29.6 ± 0.4
	Learned (Gözcü et al., 2018)	0.980 ± 0.002	30.5 ± 0.5
	Uninformed VDS (Lustig et al., 2007a)	0.959 ± 0.005	28.2 ± 0.6
Testing	Our method	0.969 ± 0.003	33.5 ± 0.9
	Learned (Gözcü et al., 2018)	0.969 ± 0.003	34.2 ± 0.7
	Uninformed VDS (Lustig et al., 2007a)	0.944 ± 0.007	31.6 ± 0.7

As we see in Figure 1.13 and Table 1.3, both our learned pattern and the learned pattern from Gözcü et al. (2018) significantly outperform the uninformed variable density sampling pattern from Lustig et al. (2007a). Our learned pattern performs very similarly to the pattern from Gözcü et al. (2018), if ever so slightly worse in terms of the performance metrics. A comparison of the computational effort required for the method in Gözcü et al. (2018) and our method can be given by noting that the effort required in both methods is proportional to the number of times a lower level problem has to be solved. In our method, there is at each iteration an additional adjoint equation that needs to be solved, which takes less than but comparable effort to one lower level solve. That is, one iteration of our method effectively requires (less than) two lower level solves. For the method in Gözcü et al. (2018), assuming a set of N candidate masks (disjoint and each of the same size) and a sampling rate r, we need to perform

$$\sum_{i=0}^{rN}(N-i) = r\left(1-\frac{r}{2}\right)N^2 + \left(1-\frac{r}{2}\right)N = \Theta(N^2).$$

lower level solves. Table 1.4 shows two concrete settings in which we compare the computational effort (in terms of effective number of lower level solves) required to use each method.

Table 1.4: A comparison of the computational efforts (measured in effective number of lower level solves) required for our method and for the method in Gözcü et al. (2018) on images of size 192×192.

	Line sampling (40.6%)	Free pattern (34.7%)
Our method	4192	6494
The method from Gözcü et al. (2018)	12087	$3.90 \cdot 10^8$

Note that we did not actually use the method in Gözcü et al. (2018) to learn a free pattern, since the number of lower level solves required to do this was prohibitive. By using a continuous optimisation approach to learning sampling patterns, our method can be more easily scaled up to higher resolutions and more computationally demanding settings such as 3D MRI or dynamic MRI; Quasi-Newton methods, such as the L-BFGS-B algorithm, exhibit a resolution independent behavior for problems like Problem (1.2) i.e., the number of outer iterations remains almost the same no matter the size of the variables involved (Kelley and Sachs, 1987).

1.3.7 High resolution example

Up to this point, the experiments have been run on relatively small images. For this experiment, we used a training set of 5 slices taken from a high resolution scan of a phantom. In Figure 1.14, we consider a different test slice from this scan to see how well the learned pattern performs. We compare our learned pattern to a low-pass sampling pattern (with the lower level regularisation parameter learned on the training set). Though both methods do well at reconstructing the phantom image, the zoomed region shows that our method allows fine details to be resolved very well, even when sampling just 5.7% of k-space, whereas the low pass pattern has a limited resolution.

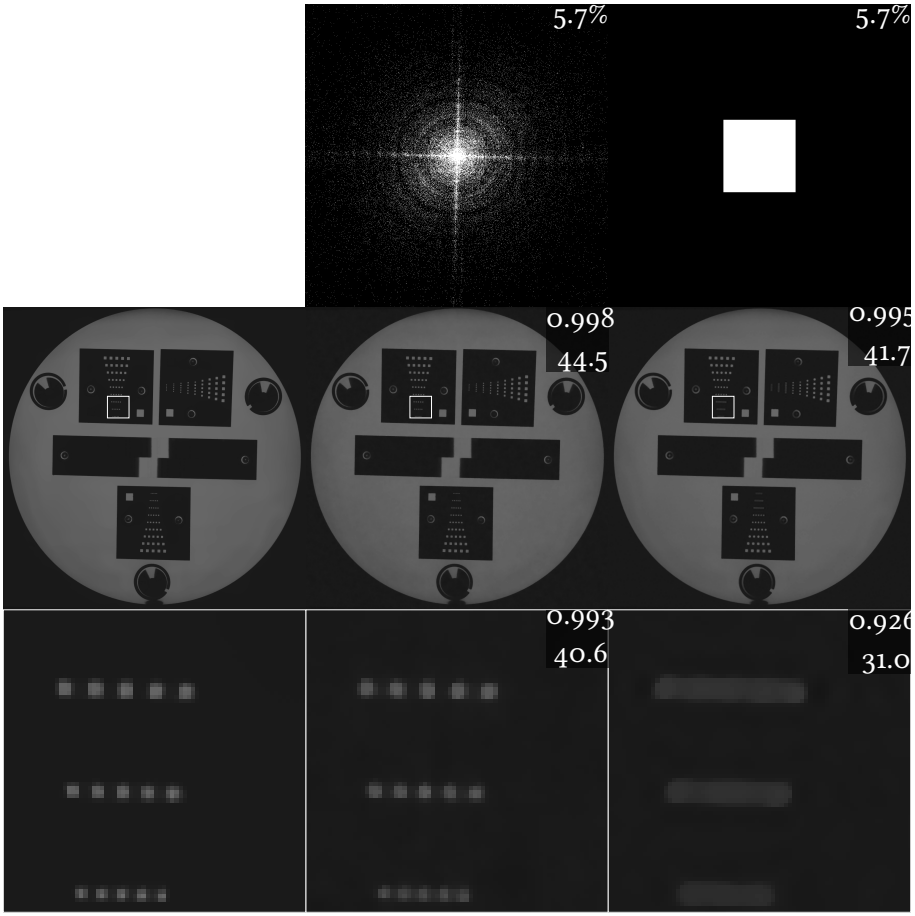

Figure 1.14: A comparison of the learned pattern and a low-pass sampling pattern in the high resolution setting with TV regularisation in the lower level problem. On each of the reconstructions, the top number is the SSIM value and the bottom number is the PSNR. On the bottom row, the performance metrics are computed using just the zoomed regions.

1.4 Conclusions and discussion

We have proposed a supervised learning approach to learn high quality sampling patterns for accelerated MRI for a given variational reconstruction method. We have demonstrated that this approach is highly flexible, allowing for a wide variety of regularisation functionals to be used and allowing constraints to be imposed on the learned sampling patterns. Furthermore, we have shown that the method can be used successfully with small training sets. The learned patterns perform favourably compared to standard choices of sampling patterns, both quantitatively (measured by SSIM and PSNR on a test set) and qualitatively (by comparing the resolution of fine scale details).

This work shows that it is feasible to learn sampling patterns by applying continuous optimisation methods to a bilevel optimisation problem. There are multiple ways in which this methodology can be extended to work in different settings.

All our experiments were carried out on 2D images. With minor mathematical modifications, the proposed method can be applied to learn sampling patterns for 3D MRI, though it is worth noting that the computational effort will scale up accordingly and the implementation will need to be optimised to deal with this. There is considerable scope for optimisation of the computational implementation of the method, for instance by parallelising the solution method for the lower level problems. To accelerate MRI in practice, it is necessary to take into account the physical constraints imposed on sampling. The free pattern of points learned by our method is not immediately useful for accelerating 2D MRI, but it can be used for accelerated 3D MRI. If our method is extended to 3D MRI, the problem of efficiently sampling along these patterns in practice comes up again. In Boyer et al. (2016), a method is proposed (which has been implemented in practice in NeuroSpin (Lazarus et al., 2019)) that can be used to generate practical sampling strategies from a given target density. We can estimate a target density from our learned pattern, and use it as an input to this method.

Besides these extensions to our method, one can consider more general lower level regularisation functionals and allow for more flexibility to learn a custom regulariser as was done for denoising in Chen (2014), or unroll the lower level algorithm and use an approach similar to that of the variational network (Hammernik et al., 2018).

In this chapter, we have considered the free and Cartesian line parametrisations of the sampling pattern, but we mentioned that any differentiable parametrisation of the sampling pattern can be used. With an appropriate choice of the parametrisation, our method can be used to learn optimal radial line patterns, or other physically feasible optimal sampling patterns.

In our framework, we made smoothness assumptions on the lower level problems in order to differentiate their solution maps. Similar results can be derived assuming partial smoothness of the regularisation functionals (Vaiter et al., 2017), which covers total variation regularisation and the wavelet regularisation without needing to smooth them. The non-smooth lower level problems will be harder to solve, but it might be possible to deal directly with non-smooth lower level problems using this approach. Alternatively, one could consider optimality conditions for bilevel optimisation problems with non-smooth lower level problems (Dempe and Zemkoho, 2011) and attempt to solve the optimality conditions.

Despite being a smooth optimisation problem, the learning procedure is computationally intensive, since the lower level problems have to be solved to high accuracy in each iteration. These issues are alleviated by warm-starting the lower level solvers and it may be possible to do something similar with the iterative solver used to compute gradients. There is considerable scope for investigating ways in which the optimisation can be improved: the problem is nonconvex so one could further research whether this is problematic in this case, and, if so, how to get around these issues. In Section 1.3.3, we saw that, even with the penalty in the upper level that encourages discreteness of the learned patterns, the learned Cartesian line patterns were not binary, which may be an artefact of the difficulties involved in solving the optimisation problem. One thing that can be of great importance in nonconvex optimisation is the initialisation that is used; in this chapter we have used a fixed initialisation consisting of an identity sampling operator and the corresponding optimal regularisation parameter and found that it generally worked well, but more detailed study may point to a more suitable initialisation. Since the objective function splits as a sum over the training set, another natural direction of future research would be to investigate the use of stochastic optimisation methods in this setting.

Appendix 1.A Alternative parametrisations of the sampling pattern

As was mentioned before, it is possible to use various parametrisations of the sampling pattern. We implement this by allowing p to depend smoothly on another parameter λ, through $p : B \to C$. This generalised parametrisation includes the following ones, which are used in the results of the main text:

- If we let $B = [0,1]^{n_1} \times [0,\infty)$ or $B = [0,1]^{n_2} \times [0,\infty)$, and we let p encode horizontal or vertical lines in k-space using the first n_1 or n_2 coordinates of λ and the regularisation parameter with the last coordinate of λ, we can learn Cartesian line patterns and the regularisation parameter,
- If we have a fixed pattern $\mathcal{S} = \operatorname{diag}(s_1, \ldots, s_{n_1 \cdot n_2})$ and let $p(\lambda) = (s_1, \ldots, s_{n_1 \cdot n_2}, \lambda)$ with $B = [0,\infty)$, we can learn the optimal regularisation parameter for the fixed pattern $\mathcal{S}$.

Instead of studying a problem like Problem (1.2), our problem now becomes

$$\min_{\lambda \in B} \frac{1}{N} \sum_{i=1}^{N} L_{u_i^*}(\hat{u}_i(p(\lambda)) + P(p(\lambda)).$$

The same methodology that is used in the main text can be used to tackle this problem and we can use the chain rule to get the gradients that we need: $\lambda \mapsto P(p(\lambda))$ has gradient given by $\nabla P_p(p(\lambda)) D_\lambda p(\lambda)$, and using Equation (1.5), we see that $\lambda \mapsto L_{u_i^*}(\hat{u}_i(p(\lambda)))$ has gradient

$$\begin{aligned} -D_\lambda p(\lambda)^* D_{p,u} E_{y_i}(\hat{u}_i(p(\lambda)); p(\lambda)) \\ ([D_u^2 E_{y_i}(\hat{u}_i(p(\lambda)); p(\lambda))]^{-1} \nabla L_{u_i^*}(\hat{u}_i(p(\lambda)))^*) \end{aligned}$$

Appendix 1.B Gradient and Hessian of the lower level regularisation

The regularisers that we consider in the lower level problems are twice continuously differentiable, and we can give explicit formulas for their gradients and for the action of their Hessians. Although we have a complex image forward model, when we speak of differentiability we mean differentiability with respect to the real and imaginary parts separately. Similarly, pixelwise products of complex quantities should here be interpreted as separate

multiplication of the real and imaginary parts. We only need to compute the gradient and Hessian of $Q(z) = Q(z^1, \dots, z^M) = \sum_i \rho(|(z^1, \dots, z^M)|)$. Indeed, the regulariser J satisfies $J(u) = Q(\mathcal{A}u)$, so $D_u J(u) = \mathcal{A}^* D_z Q(\mathcal{A}u)$ and $D_u^2 J(u) = \mathcal{A}^* D_z^2 Q(\mathcal{A}u)\mathcal{A}$. We denote the real and imaginary parts of z^j by z^j_{real} and z^j_{imag} respectively. Differentiating the sum that defines Q with respect to $z^j_{\text{real},i}, z^j_{\text{imag},i}$, we find that

$$\frac{\partial Q}{\partial z^j_{\text{comp},i}}(z) = \frac{\rho'(|z|_i)}{|z|_i} z^j_{\text{comp},i}, \qquad \text{for comp} \in \{\text{real}, \text{imag}\}. \tag{1.7}$$

We make notation less cumbersome by defining $\phi(x) = \rho'(x)/x$. Using Expression (1.7), we see that

$$D_z Q(z) = \phi(|z|) \cdot z. \tag{1.8}$$

To get the Hessian of Q, consider a component $(D_z Q(z))^p_{\text{comp},i}$ and differentiate with respect to $z^q_{\text{comp'},j}$:

$$\begin{aligned}\frac{\partial^2 Q}{\partial z^q_{\text{comp'},j} \partial z^p_{\text{comp},i}}(z) = \frac{\phi'(|z|_i)}{|z|_i} \delta_{i,j} z^q_{\text{comp'},j} z^p_{\text{comp},i} \\ + \phi(|z|_i) \delta_{(i,p,\text{comp}),(j,q,\text{comp'})}.\end{aligned} \tag{1.9}$$

For ease of notation, we define

$$\psi(x) = \begin{cases} 0 & \text{if} \quad x = 0 \\ \frac{\phi'(x)}{x} & \text{if} \quad x > 0. \end{cases}.$$

The action of $D^2 Q(z)$ on a vector w can now be computed using Equation (1.9):

$$\begin{aligned}D_z^2 Q(z) w = \psi(|z|) \cdot z \cdot \Bigg(\sum_{\substack{p=1,\dots,M \\ \text{comp} \in \{\text{real},\text{imag}\}}} z^p_{\text{comp}} \cdot w^p_{\text{comp}} \Bigg) \\ + \phi(|z|) \cdot w\end{aligned} \tag{1.10}$$

Appendix 1.C Details of solving the lower level problems

In Section 1.2.3 of the main text, we show that the lower level energy functional E_y takes the saddle-point structure that is exploited in PDHG. In this section, we describe the computations that need to be made to choose the parameters correctly and apply the algorithm.

1.C.1 Proximal operator of F_2

Given how F_2 is defined, its proximal operator can be computed by applying pixelwise the proximal operator of $\xi : x = (x^1, \dots, x^M) \mapsto \alpha(p)\rho(\sqrt{|x^1|^2 + \dots + |x^M|^2})$. The optimality condition defining the proximal operator tells us that $\operatorname{prox}_{\tau\xi}(x^1, \dots, x^M)$ is the unique $\hat{x}$ satisfying

$$(1 + \tau\alpha(p)\phi(|\hat{x}|))\hat{x} = x.$$

That is, $\hat{x}$ is a scalar multiple of x. Taking norms of both sides of this equation, we get an equation

$$(1 + \tau\alpha(p)\phi(C))C = |x|,$$

which is explicitly solvable for our choices of lower level regularisations, for $|\hat{x}|$ in terms of $|x|$. Denoting its solution by $C(|x|, \tau)$, we find that $\operatorname{prox}_{\tau\xi}(x) = \hat{x} = C(|x|, \tau)x/|x|$, and hence $\operatorname{prox}_{\tau F_2}(z)_i = \operatorname{prox}_{\tau\xi}(z_i) = C(|z_i|, \tau)z_i/|z_i|$.

1.C.2 Choosing the parameters and putting the algorithm together

To apply PDHG, we need to be able to compute proximal operators for F^* and G. Since Moreau's identity gives an explicit expression relating the proximal operator of F and of F^*, it suffices to compute the proximal operator of F. Furthermore, since F is separable, we have $\operatorname{prox}_{\tau F}(v_1, v_2) = (\operatorname{prox}_{\tau F_1}(v_1), \operatorname{prox}_{\tau F_2}(v_2))$. In the previous subsection, we showed that we can explicitly compute $\operatorname{prox}_{\tau F_2}$. Considering the optimality condition defining $\operatorname{prox}_{\tau F_1}$ we find that

$$\operatorname{prox}_{\tau F_1}(v) = \mathcal{F}^{-1}(I + \tau\mathcal{S}(p)^2)^{-1}(\mathcal{F}u + \tau\mathcal{S}(p)y). \tag{1.11}$$

Note that $I + \tau\mathcal{S}(p)^2$ is a diagonal matrix so that its inverse can be computed by a simple coordinate-wise product between vectors. Since $G(u) = \varepsilon\|u\|^2/2$, we have $\operatorname{prox}_{\tau G}(u) = u/(\varepsilon\tau + 1)$.

To choose appropriate step sizes, we note that F is strongly smooth, since F_1 is (its Hessian is $\mathcal{F}^{-1}\mathcal{S}(p)^2\mathcal{F}$, the norm of which is bounded above by $\|\mathcal{S}(p)^2\| = \max_{i=1,\dots,n} p_i^2$) and F_2 is as well (with constant bounded by $c(p)$ as shown in Section 1.C.3). Hence the smoothness constant of F is bounded by $\eta := \max\{\max_{i=1,\dots,n} p_i^2, c(p)\}$. Furthermore, G is strongly convex with constant ε. Finally, we need an estimate on $\|\mathcal{K}\|$: since $\mathcal{K} = (I, \mathcal{A})$, we have $\|\mathcal{K}\| = \sqrt{1 + \|\mathcal{A}\|^2}$. In the examples we consider, $\|\mathcal{A}\|$ is known or can be estimated from above: when $\mathcal{A} = W$ is an orthogonal wavelet transform we have $\|\mathcal{A}\| = 1$, while when $\mathcal{A} = \nabla$ we discretise the gradient operator using first-order forward differences with zero Neumann boundary conditions, for which it can be shown that $\|\mathcal{A}\| \leqslant \sqrt{8}$ (Chambolle, 2004). Indeed, $\mathcal{A}u = (\partial_x u, \partial_y u)$, where $\partial_x : \mathbf{C}^n \to \mathbf{C}^n$ and $\partial_y : \mathbf{C}^n \to \mathbf{C}^n$ are the linear operators computing differences in the directions along the rows and columns on the flattened images:

$$(\partial_x u)_i = (u_{i+n_1} - u_i) \cdot \mathbf{1}_{i \leqslant n_1(n_2-1)}, \qquad (\partial_y u)_i = (u_{i+1} - u_i) \cdot \mathbf{1}_{i \not\equiv 0 \bmod n_1}.$$

Hence, using the triangle inequality and Young's inequality, we find

$$\begin{aligned}
\|\partial_x u\|^2 &= \sum_{1 \leqslant n_1(n_2-1)} |u_{i+n_1} - u_i|^2 \\
&\leqslant \sum_{1 \leqslant i \leqslant n_1(n_2-1)} (|u_{i+n_1}|^2 + 2|u_{i+n_1}||u_i| + |u_i|^2) \\
&\leqslant 2 \sum_{1 \leqslant i \leqslant n_1(n_2-1)} (|u_{i+n_1}|^2 + |u_i|^2) \leqslant 4\|u\|^2.
\end{aligned}$$

In the exact same fashion, we can show that $\|\partial_y u\|^2 \leqslant 4\|u\|^2$, so $\|\mathcal{A}u\|^2 = \|\partial_x u\|^2 + \|\partial_y u\|^2 \leqslant 8\|u\|^2$ as desired.

In any case, we have $\|\mathcal{A}\| \leqslant L$ for some known $L > 0$. Choosing our parameters as

$$\mu = 2\sqrt{\frac{\varepsilon}{(1+L^2)\eta}}, \quad \tau = \frac{\mu}{2\varepsilon}, \quad \sigma = \frac{\mu\eta}{2}, \quad \theta = \frac{1}{1+\mu},$$

makes PDHG converge linearly (Chambolle and Pock, 2011).

1.C.3 Computing the smoothness constant of F_2 for solving the lower level problems

To compute step sizes for PDHG that give a linearly convergent algorithm, we require an estimate of the smoothness constant of F_2. Recall that F_2 can be written as $F_2(z) = \alpha(p)J(z)$.

The smoothness constant of J can be estimated by an upper bound on the operator norm of the Hessian. Using the triangle inequality, Equation (1.10) tells us that

$$\|D_z^2 J(z)w\| \leqslant \sum_{\substack{p=1,\dots,M \\ \text{comp}\in\{\text{real,imag}\}}} \left\|\psi(|z|)\cdot z\cdot\left(z^p_{\text{comp}}\cdot w^p_{\text{comp}}\right)\right\| + \|\phi(|z|)\cdot w\|. \tag{1.12}$$

Let us consider a term with index (p, comp) in the first sum:

$$\left(\psi(|z|)\cdot z\cdot(z^p_{\text{comp}}\cdot w^p_{\text{comp}})\right)^q_{\text{comp'},i} = \psi(|z|_i) z^q_{\text{comp'},i} z^p_{\text{comp},i} w^p_{\text{comp},i}.$$

Since $|z^q_{\text{comp'},i} z^p_{\text{comp},i}| \leqslant \frac{1}{2}(|z^q_{\text{comp'},i}|^2 + |z^q_{\text{comp},i}|^2) \leqslant \frac{1}{2}|z|_i^2$, we find that

$$|\psi(|z|_i)\cdot z_i\cdot(z^p_{\text{comp},i}\cdot w^p_{\text{comp},i})| \leqslant \frac{1}{2}\sup_{x\geqslant 0}(|\psi(x)|x^2)|w^p_{\text{comp},i}|.$$

Now $|w^p_{\text{comp, i}}| \leqslant |w|_i$ and $\psi(x)x = \phi'(x)$, so we conclude that

$$\left\|\psi(|z|)\cdot z\cdot\left(z^p_{\text{comp}}\cdot w^p_{\text{comp}}\right)\right\| \leqslant \frac{\sqrt{2M}}{2}\sup_{x\geqslant 0}(|\phi'(x)|x)\|w\|. \tag{1.13}$$

For the final term in Inequality (1.12), we can simply use the bound

$$\|\phi(|z|)\cdot w\| \leqslant \sup_{x\geqslant 0}|\phi(x)|\|w\|. \tag{1.14}$$

Combining the above inequalities, we find that

$$\|D_z^2 J(z)\| \leqslant \sqrt{2}M^{\frac{3}{2}}\sup_{x\geqslant 0}(|\phi'(x)|x) + \sup_{x\geqslant 0}|\phi(x)|, \tag{1.15}$$

so the functional J is L-smooth with

$$L = \sqrt{2}M^{\frac{3}{2}}\sup_{x\geqslant 0}(|\phi'(x)|x) + \sup_{x\geqslant 0}|\phi(x)|$$

and $F_2 = \alpha(p)J$ has smoothness constant bounded by $c(p) = \alpha(p)L$.

Appendix 1.D Computing the action of the Hessian of the lower level energy functional

In this section, we compute the action of the Hessian of the lower level energy functionals. To prevent the expressions from becoming overly cumbersome, let us split E into parts:

$$E_y(u;p) = E_{\text{data}}(u;p) + E_{\text{reg}}(u;p) + E_{\varepsilon-\text{convex}}(u;p),$$

with

$$\begin{aligned} E_{\text{data}}(u;p) &= \frac{1}{2}\|\mathcal{S}(p)(\mathcal{F}u - y)\|^2, \\ E_{\text{reg}}(u;p) &= \alpha(p)J(\mathcal{A}u), \\ E_{\varepsilon-\text{convex}}(u;p) &= \frac{\varepsilon}{2}\|u\|^2. \end{aligned}$$

We can differentiate each of these components with respect to u (using the results shown in Section 1.B to differentiate E_{reg}) to give

$$\begin{aligned} D_u E_{\text{data}}(u;p) &= \mathcal{F}^{-1}\mathcal{S}(p)^2(\mathcal{F}u - y), \\ D_u E_{\text{reg}}(u;p) &= \alpha(p)\mathcal{A}^*(\phi(|\mathcal{A}u|)\cdot\mathcal{A}u), \\ D_u E_{\varepsilon-\text{convex}}(u;p) &= \varepsilon u. \end{aligned}$$

Differentiating once again with respect to u (again using the results in Section 1.B), we find that the actions of the various parts of the Hessian on a vector w are given by

$$\begin{aligned} D_u^2 E_{\text{data}}(u;p)w &= \mathcal{F}^{-1}\mathcal{S}(p)^2\mathcal{F}w, \\ D_u^2 E_{\text{reg}}(u;p)w &= \alpha(p)\cdot\mathcal{A}^*\Bigg(\psi(|\mathcal{A}u|)\cdot\mathcal{A}u\cdot \\ &\qquad \Bigg(\sum_{\substack{p=1,\dots,M \\ \text{comp}\in\{\text{real, imag}\}}} (\mathcal{A}u)^p_{\text{comp}}\cdot(\mathcal{A}w)^p_{\text{comp}}\Bigg) \\ &\qquad + \phi(|\mathcal{A}u|)\cdot\mathcal{A}w\Bigg), \\ D_u^2 E_{\varepsilon-\text{convex}}(u;p)w &= \varepsilon w. \end{aligned}$$

In addition to this, according to Equation (1.5), we need access to $D_{p,u}$. Noting that $E_{\varepsilon-\text{convex}}$ does not depend on p, we find that $D_{p,u}E_y$ acts on a vector w as

$$(D_{p,u}E_y(u;p)w)_i = \sum_{\text{comp}\in\{\text{real,imag}\}} (\mathcal{F}w)_{\text{comp},i} \cdot 2p_i \cdot (\mathcal{F}u - y)_{\text{comp},i},$$

for $1 \leqslant i \leqslant n$ (for the components of p corresponding to the points in the sampling pattern), and (for the component of p corresponding to the lower level regularisation parameter)

$$(D_{p,u}E_y(u;p)w)_{n+1} = w^*\mathcal{A}^*(\phi(|\mathcal{A}u|) \cdot \mathcal{A}u).$$

Chapter 2

Equivariant neural networks for inverse problems

2.1 Introduction

In the previous chapter, we considered how bilevel optimisation problems can be posed and solved to learn from data how to better solve inverse problems. There are two main issues that one may have with such an approach:

- We consider lower level problems that still rely on hand-crafted regularisation functionals, which encode relatively crude prior information about the structure that we desire the solution to have. Depending on the data that we are considering, these hand-crafted regularisation functionals may be wildly inappropriate.

- The solution of the bilevel optimisation problem requires a vast amount of computational effort: there is an inner loop, in which hundreds of lower level iterations are required to get lower level solutions and gradients that are accurate enough, as shown in Figure 1.2 and Figure 1.3.

The first issue need not be too much of a problem; we can choose to parametrise the regularisation functional in a more flexible way, for example using neural networks. In fact, we can even do so while retaining convexity of the regularisation functional (which ensures that we can be confident in solving the lower level problem), by using input-convex neural networks (Amos et al., 2017). The second issue seems to be more persistent, and has prompted research into other ways in which machine learning can be used to solve inverse problems.

In this chapter, we will be particularly interested in so-called learned iterative reconstruction methods, which we previously mentioned in the introduction chapter. We will discuss how extra structure, in the form of symmetries, can naturally be incorporated into such methods and the benefits that come with this approach.

Let it be noted that awareness of symmetries has been used to great effect in deep learning in the past. Convolutional neural networks (CNNs) (LeCun and Bengio, 1998) are a standard tool in deep learning methods for images. By learning convolutional filters, CNNs naturally encode translational symmetries of images: if τ_h is a translation by $h \in \mathbf{R}^d$, and k, f are functions on $\mathbf{R}^d$, we formally have the following relation (translational equivariance)

$$\tau_h[k * f] = k * [\tau_h f]. \tag{2.1}$$

This allows learned feature detectors to detect features regardless of their position (though not their orientation or scale) in an image. In many cases it may be desirable for these learned feature detectors to also work when images are transformed under other group transformations, i.e. one may ask that a property such as Equation (2.1) holds for a more general group transformation than the group of translations $\{\tau_h | h \in \mathbf{R}^d\}$. If natural symmetries of the problem are not built into the machine learning method and are not present in the training data, in the worst case, it can result in catastrophic failure as illustrated in Figure 2.1.

To some extent, this problem is circumvented by augmenting the training data through suitable transformations, but it has been shown in classification and segmentation tasks that it is still beneficial to incorporate known symmetries directly into the architecture used, especially if the amount of training data is small (Bekkers et al., 2018; Weiler and Cesa, 2019; Worrall et al., 2017). Furthermore, training on augmented data is not enough to guarantee that the final model satisfies the desired symmetries. There has recently been a considerable amount of work in this direction, in the form of group equivariant CNNs. Most of the focus has been on roto-translational symmetries of images (Bekkers et al., 2018; Cohen and Welling, 2016; Dieleman et al., 2016; Weiler and Cesa, 2019) (when the group is the so-called Euclidean group), though there is also some work on incorporating scaling symmetries (Sosnovik et al., 2019; Worrall and Welling, 2019) and even on equivariance to arbitrary Lie group symmetries (Finzi et al., 2020).

As mentioned before, we will concern ourselves with solving inverse imaging problems: given measurements y that are related to an underlying ground truth image u^* through a model

$$y = \mathfrak{N}(A(u^*)), \tag{2.2}$$

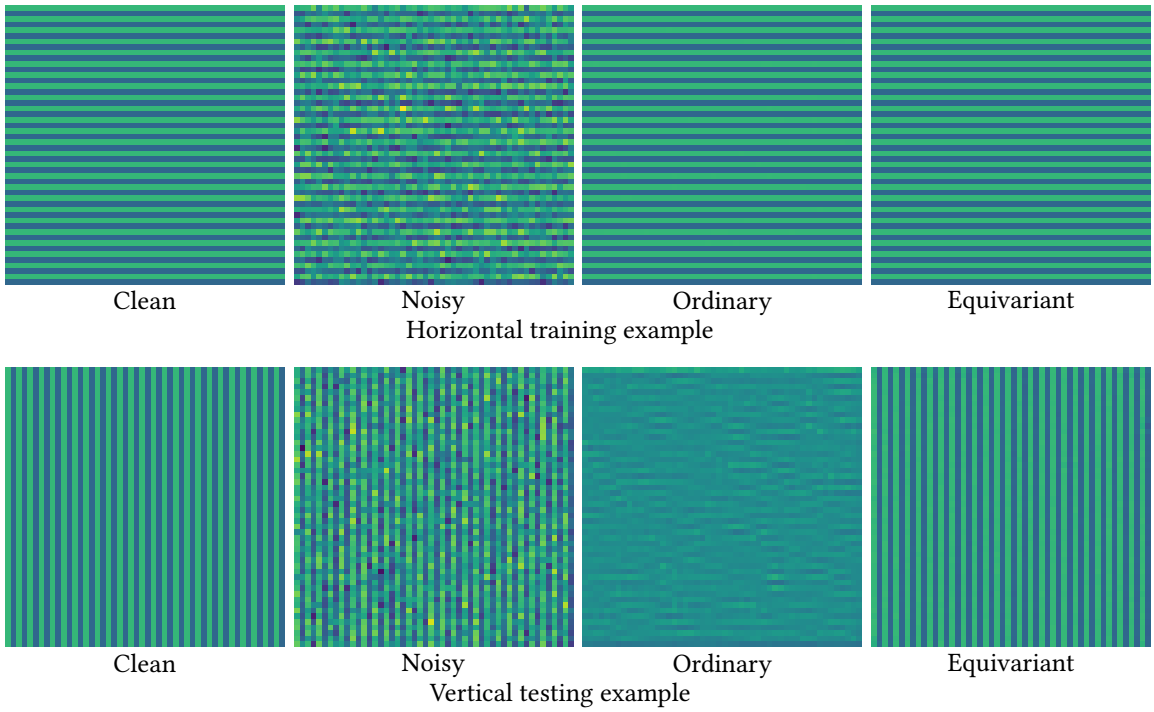

Figure 2.1: Roto-translationally ('Equivariant') and just translationally ('Ordinary') equivariant filters are trained to denoise on a single pair of ground truth and noisy images ('Clean' and 'Noisy' in the top row), giving perfect denoising results on the training example. In the bottom row, we see the result of testing the learned filters on a rotated version of the training image; the ordinary filter completely fails at recovering the ground truth, whereas the equivariant filter performs as well as it did on the training image. All images are displayed using the same colour range.

with A the so-called forward operator and $\mathfrak{N}$ a noise-generating process, the goal is to estimate the image u from the measurements y as well as possible. Typical examples of inverse imaging problems include the problem of recovering an image from its line integrals as in computerised tomography (CT) (Hounsfield, 1973), or recovering an image from subsampled Fourier measurements as in magnetic resonance imaging (MRI) (Lauterbur, 1973; Mansfield and Grannell, 1975). The solution of an inverse problem is often complicated by the presence of ill-posedness: a problem is said to be well-posed in the sense of Hadamard (Hadamard, 1902) if it satisfies a set of three conditions (existence of a solution, its uniqueness, and its continuous dependence on the measurements), and ill-posed if any of these conditions fail.

It is a natural idea to try to apply equivariant neural networks to solve inverse imaging problems: there is useful knowledge about the relationship between a ground truth image and its measurements in the form of A and the symmetries in both the measurement and image domain (the range and domain of A respectively). Furthermore, training data tends to

be considerably less abundant in medical and scientific imaging than in the computer vision and image analysis tasks that are typical of the deep learning revolution, such as ImageNet classification (Krizhevsky et al., 2012). This suggests that the lower sample complexity of equivariant neural networks (as compared to ordinary CNNs) may be harnessed in this setting with scarce data to learn better reconstruction methods. Finally, end users of the methods, e.g. medical practitioners, are often skeptical of "black-box" methods and guarantees on the behaviour of the method, such as equivariance of the method to certain natural image transformations, may alleviate some of the concerns that they have.

We investigate the use of equivariant neural networks within the framework of learned iterative reconstruction methods (Adler and Öktem, 2017; Putzky and Welling, 2017), which constitute some of the most prototypical deep learning solutions to inverse problems. The designs of these methods are motivated by classical variational regularisation approaches (Engl et al., 1996; Hansen, 2010), which propose to overcome the ill-posedness of an inverse problem by estimating its solution as

$$\hat{u} = \operatorname*{argmin}_u d(A(u), y) + J(u), \tag{2.3}$$

with d a measure of discrepancy motivated by our knowledge of the noise-generating process $\mathfrak{N}$ and J is a regularisation functional incorporating prior knowledge of the true solution. Learned iterative reconstruction methods, also known as unrolled iterative methods, are designed by starting from a problem such as Problem (2.3), choosing an iterative optimisation method to solve it, truncating that method to a finite number of iterations, and finally replacing parts of it (e.g. the proximal operators) by neural networks. We will show that these neural networks can naturally be chosen to be equivariant neural networks, and that doing so gives improved performance over choosing them to be ordinary CNNs. More precisely, our contributions in this chapter are as follows:

Our contributions

We show that invariance of a functional to a group symmetry implies that its proximal operator satisfies an equivariance property with respect to that group. This insight can be combined with the unrolled iterative method approach: it makes sense for a regularisation functional to be invariant to roto-translations if there is no prior knowledge on the orientation and position of structures in the images, in which case the corresponding proximal operators are roto-translationally equivariant.

Motivated by these observations, we build learned iterative reconstruction methods using roto-translationally equivariant building blocks. We show in a supervised learning setting that these methods can outperform comparable methods that only use ordinary convolutions as building blocks, when applied to a low-dose CT reconstruction problem and a subsampled MRI reconstruction problem. This superior performance is manifested in two main ways: the equivariant method is better able to take advantage of small training sets than the ordinary one, and its performance is more robust to transformations that leave images in orientations not seen during training.

2.2 Notation and background

In this section, we give an overview of the main concepts regarding groups and representations that are required to follow the main text. By a group G, we mean a set equipped with an associative binary operation $\cdot : G \times G \to G$ (usually the dot is omitted in writing), furthermore containing a neutral element e, such that $e \cdot g = g \cdot e = g$ for all $g \in G$ and a unique inverse g^{-1} for each group element g, such that $g \cdot g^{-1} = g^{-1} \cdot g = e$. Given groups G and H, we say that a map $\phi : G \to H$ is a group homomorphism if it respects the group structures:

$$\phi(g_1 g_2) = \phi(g_1)\phi(g_2) \quad \text{for any} \quad g_1, g_2 \in G.$$

Groups can be naturally used to describe symmetries of mathematical objects through the concept of group actions. Given a group G and set X, we say that G acts on X if there is a function $T : G \times X \to X$ (the application of which we stylise as $T_g[x]$ for $g \in G, x \in X$) that obeys the group structure in the sense that

$$T_{g_1} \circ T_{g_2} = T_{g_1 g_2} \quad \text{for any} \quad g_1, g_2 \in G \tag{2.4}$$

and $T_e = \mathrm{id}$. That is, the group action can be thought of as a group homomorphism from G to the permutation group of X. If there is no ambiguity, the group action may just be written as $T_g[x] = g \cdot x = gx$. An important type of group actions is given by the group representations. If V is a vector space, we will denote by $\mathrm{GL}(V)$ its general linear group, the group of invertible linear maps $V \to V$, with the group operation given by composition. A representation $\rho : G \to \mathrm{GL}(V)$ of a group G which acts on V is a group homomorphism, and so corresponds to a linear group action T of G on V: $\rho(g)x = T_g[x]$ for $x \in V$ and $g \in G$. Given a vector space V, any group G has a representation on V given by $\rho(g) = I$, which is the so-called trivial representation. If V is additionally a Hilbert space, we will call ρ a unitary representation if $\rho(g)$ is a unitary operator for each $g \in G$, i.e. $\|\rho(g)x\| = \|x\|$ for all $x \in V$. Given a finite group $G = \{g_1, \ldots, g_n\}$, we can define the so-called regular representation ρ of G on $\mathbf{R}^n$ by

$$\rho(g_i)e_j = e_k,$$

where $\{e_1, \ldots, e_n\}$ is a basis of $\mathbf{R}^n$ and k is such that $g_i g_j = g_k$. With this representation, each $\rho(g)$ is a permutation matrix, so ρ is a unitary representation if the basis $\{e_1, \ldots, e_n\}$ is orthonormal.

In this work, the groups that we will consider take the form of a group of isometries on $\mathbf{R}^d$. These groups are represented by a semi-direct product $G = \mathbf{R}^d \rtimes H$, where H is a subgroup of the orthogonal group $\mathrm{O}(d)$ of rotations and reflections:

$$\mathrm{O}(d) = \{R \in \mathrm{GL}(\mathbf{R}^d) | R^T = R^{-1}\}.$$

That is to say, the groups we consider are subgroups of the Euclidean group $\mathrm{E}(d) = \mathbf{R}^d \rtimes \mathrm{O}(d)$, which is a quintessential example of a Lie group (Hall, 2015): a smooth manifold equipped with a group structure that is compatible with the manifold, in the sense that the group product and inverse are smooth functions. We will not explicitly use the Lie group structure in what follows.

An important subgroup of $\mathrm{O}(d)$ is the special orthogonal group $\mathrm{SO}(d) = \{A \in \mathrm{O}(d) | \det(A) = 1\}$, which represents the set of pure rotations in $\mathrm{O}(d)$. Each element of the semi-direct product G can be identified with a unique pair (t, R) of $t \in \mathbf{R}^d$, the translation component, and $R \in H$, the rotation (and potentially reflection). The semi-direct product can naturally be encoded as a matrix using homogeneous coordinates

$$(t, R) \leftrightarrow \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix},$$

by which we mean that the group product can be thought of as a matrix product:

$$(t, R) \cdot (t', R') \leftrightarrow \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R' & t' \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} RR' & Rt' + t \\ 0 & 1 \end{pmatrix} \leftrightarrow (Rt' + t, RR').$$

Note in particular that this group product $(t, R) \cdot (t', R') = (Rt' + t, RR')$ is not the ordinary (direct) product; there is an additional "twist". G naturally acts on a point $x \in \mathbf{R}^d$ through $T_{(t,R)}[x] = (t, R)x = Rx + t$.

In the experiments that we consider later in this work, we will consider the case $d = 2$. In this case $\mathrm{SO}(d)$ has a simple representation:

$$\mathrm{SO}(2) = \left\{ \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \middle| \theta \in [0, 2\pi) \right\}.$$

We will identify the groups $\mathbf{Z}_m$ of integers modulo m, also known as the cyclic group of order m, with the subgroup of $\mathrm{SO}(2)$ given by

$$\mathbf{Z}_m = \left\{ \begin{pmatrix} \cos(2\pi k/m) & -\sin(2\pi k/m) \\ \sin(2\pi k/m) & \cos(2\pi k/m) \end{pmatrix} \middle| k \in \mathbf{Z} \right\}.$$

Given vector spaces V_1, V_2, we will denote by $\mathrm{Hom}(V_1, V_2)$ the vector space of linear operators $A : V_1 \to V_2$. We will refer to a number of function spaces: $L^2(\mathbf{R}^d, \mathbf{R}^c)$ denotes the Hilbert space of square integrable functions $f : \mathbf{R}^d \to \mathbf{R}^c$ (where $\mathbf{R}^c$ carries the Euclidean norm), identified as usual up to equality almost everywhere, and $C_c^\infty(\mathbf{R}^d, \mathbf{R}^c)$ denotes the vector space of infinitely smooth functions $f : \mathbf{R}^d \to \mathbf{R}^c$ that have compact support.

2.3 Learnable equivariant maps

The concept of equivariance is well-suited to describing the group symmetries that a function might obey:

Definition 2.3.1. Given a general group G, a function $\Phi : \mathcal{X} \to \mathcal{Y}$ and group actions $T^{\mathcal{X}}, T^{\mathcal{Y}}$ of G on $\mathcal{X}$ and $\mathcal{Y}$ respectively, Φ will be called equivariant if it satisfies

$$\Phi(T_g^{\mathcal{X}}[f]) = T_g^{\mathcal{Y}}[\Phi(f)] \tag{2.5}$$

for all $f \in \mathcal{X}$ and $g \in G$.

Following the definition of equivariance, we see that equivariant functions have the convenient property that composing them results in an equivariant function, as long as the group actions on the inputs and outputs match in the appropriate way:

Lemma 2.3.1. *Suppose that G is a group that acts on sets $\mathcal{X}, \mathcal{Y}$ and $\mathcal{Z}$ through $T^{\mathcal{X}}, T^{\mathcal{Y}}$ and $T^{\mathcal{Z}}$. If $\Phi : \mathcal{X} \to \mathcal{Y}$ and $\Psi : \mathcal{Y} \to \mathcal{Z}$ are equivariant, then so is $\Psi \circ \Phi : \mathcal{X} \to \mathcal{Z}$.*

Based on this property it is clear that the standard approach to building neural networks (compose linear and nonlinear functions with learnable components in an alternating manner) can be used to build equivariant neural networks as long as linear and nonlinear functions with the desired equivariance can be constructed.

Example 2.3.1. Suppose that $\mathcal{X} = L^2(\mathbf{R}^d, \mathbf{R}^{c_{\mathcal{X}}})$ and $\mathcal{Y} = L^2(\mathbf{R}^d, \mathbf{R}^{c_{\mathcal{Y}}})$, with the group $G = \mathbf{R}^d$ acting on $\mathcal{X}$ by $T_h^{\mathcal{X}}[f](x) = f(x - h)$, and in a similar way on $\mathcal{Y}$ by $T_h^{\mathcal{Y}}[f](x) = f(x - h)$. Ordinary CNNs (LeCun and Bengio, 1998), with convolutional linear layers and pointwise nonlinear functions, are equivariant in this setting.

In this work, we will consider the group $G = \mathbf{R}^d \rtimes H$ for some subgroup H of $\mathrm{O}(d)$ (see Section 2.2 for some background), acting on vector-valued functions. To be more specific, we will let $\mathcal{X} = L^2(\mathbf{R}^d, \mathbf{R}^{d_{\mathcal{X}}})$ be the Hilbert space of square-integrable $\mathbf{R}^{d_{\mathcal{X}}}$-valued functions and assume that $\mathbf{R}^{d_{\mathcal{X}}}$ carries a representation $\pi_{\mathcal{X}} : H \to \mathrm{GL}(\mathbf{R}^{d_{\mathcal{X}}})$. Similarly, we will define $\mathcal{Y} = L^2(\mathbf{R}^d, \mathbf{R}^{d_{\mathcal{Y}}})$ and assume that $\pi_{\mathcal{Y}} : H \to \mathrm{GL}(\mathbf{R}^{d_{\mathcal{Y}}})$ is a representation of H. We define the group actions $T^{\mathcal{X}}$ and $T^{\mathcal{Y}}$ to be the induced representations, $\rho_{\mathcal{X}}$ and $\rho_{\mathcal{Y}}$, of $\pi_{\mathcal{X}}$ and $\pi_{\mathcal{Y}}$ on $\mathcal{X}$ and $\mathcal{Y}$ respectively. In the setting that we are considering, these representations take a particularly simple form. As mentioned in Section 2.2, since we assume that G takes the semi-direct product form $\mathbf{R}^d \rtimes H$, each group element $g \in G$ can be uniquely thought of as a

pair $g = (t, R)$ for some $t \in \mathbf{R}^d$ and $R \in H$. With this in mind, the representations $\rho_{\mathcal{X}}$ and $\rho_{\mathcal{Y}}$ can be written as follows for any $f \in \mathcal{Z}, x \in \mathbf{R}^d$ and $t \in \mathbf{R}^d, R \in H$:

$$\rho_{\mathcal{Z}}((t,R))[f](x) = \underbrace{\pi_{\mathcal{Z}}(R)}_{\text{(a)}} \underbrace{f((t,R)^{-1}x)}_{\text{(b)}} \quad \text{for} \quad \mathcal{Z} = \mathcal{X}, \text{ or } \mathcal{Z} = \mathcal{Y}. \tag{2.6}$$

These representations have a natural interpretation: to apply a group element (t, R) to a vector-valued function, we must move the vectors, as in part (b) of Equation (2.6), and transform each vector accordingly, as in part (a) of Equation (2.6).

2.3.1 Equivariant linear operators

It is well-established that equivariant linear operators are strongly connected to the concept of convolutions. Indeed, in a relatively general setting it has been shown that an integral operator is equivariant if and only if it is given by a convolution with an appropriately constrained kernel (Cohen et al., 2019). In the setting that we are considering, the more specific result in Proposition 2.3.1 can be derived, as done in Weiler and Cesa (2019); Weiler et al. (2018b) for the case $d = 2$ and Weiler et al. (2018a) for the case $d = 3$.

Proposition 2.3.1. *Suppose that $\Phi : \mathcal{X} \to \mathcal{Y}$ is an operator given by integration against a continuous kernel $K : \mathbf{R}^d \times \mathbf{R}^d \to \mathrm{Hom}(\mathbf{R}^{d_{\mathcal{X}}}, \mathbf{R}^{d_{\mathcal{Y}}})$,*

$$\Phi(f)(x) = \int_{\mathbf{R}^d} K(x,y) f(y)\, \mathrm{d}y.$$

Then the operator Φ is equivariant if and only if it is in fact given by a convolution satisfying an additional constraint: there is a continuous $k : \mathbf{R}^d \to \mathrm{Hom}(\mathbf{R}^{d_{\mathcal{X}}}, \mathbf{R}^{d_{\mathcal{Y}}})$

$$\Phi(f)(x) = \int_{\mathbf{R}^d} k(x-y) f(y)\, \mathrm{d}y,$$

where k satisfies the additional condition

$$k(Rx) = \pi_{\mathcal{Y}}(R) k(x) \pi_{\mathcal{X}}(R^{-1}) \quad \textit{for} \quad x \in \mathbf{R}^d, R \in H.$$

The derivation of this result proceeds by writing out the definitions of equivariance and using the invariances of the Lebesgue measure. The equivariance of Φ implies that we have

the following chain of equalities for any $x \in \mathbf{R}^d$, $f \in \mathcal{X}$, $t \in \mathbf{R}^d$, $R \in H$ and $g = (t, R) \in G$:

$$\begin{aligned}
\int_{\mathbf{R}^d} \pi_{\mathcal{Y}}(R) K(g^{-1}x, y) f(y)\,\mathrm{d}y &\overset{(a)}{=} \pi_{\mathcal{Y}}(R) \int_{\mathbf{R}^d} K(g^{-1}x, y) f(x)\,\mathrm{d}y \\
&= \rho_{\mathcal{Y}}(g)[\Phi(f)](x) \\
&\overset{(b)}{=} \Phi(\rho_{\mathcal{X}}(g)[f])(x) \\
&= \int_{\mathbf{R}^d} K(x, y) \rho_{\mathcal{X}} g[f](y)\,\mathrm{d}y \\
&= \int_{\mathbf{R}^d} K(x, y) \pi_{\mathcal{X}}(h) f(g^{-1}y)\,\mathrm{d}y \\
&\overset{(c)}{=} \int_{\mathbf{R}^d} K(x, gy) \pi_{\mathcal{X}}(h) f(y)\,\mathrm{d}y.
\end{aligned}$$

Here the tags above the equality signs correspond to the following justifications:

(a) Since $\pi_{\mathcal{Y}}$ is a group representation, $\pi_{\mathcal{Y}}(R)$ is a linear map and commutes with the integral,

(b) Φ is assumed to be equivariant,

(c) We make the substitution $y \leftarrow gy$ and note that the Lebesgue measure is invariant to G.

Taking the left-hand side and right-hand side together, we find that

$$\int_{\mathbf{R}^d} \Big(\pi_{\mathcal{Y}}(R) K(g^{-1}x, y) - K(x, gy) \pi_{\mathcal{X}}(R) \Big) f(y)\,\mathrm{d}y = 0,$$

and since this must hold for any $f \in \mathcal{X} = L^2(\mathbf{R}^d, \mathbf{R}^{d_{\mathcal{X}}})$, we conclude by testing on sequences converging to Dirac delta functions that

$$\pi_{\mathcal{Y}}(R) K(g^{-1}x, y) = K(x, gy) \pi_{\mathcal{X}}(R). \tag{2.7}$$

Specialising by setting R equal to the identity element, we see that

$$K(x - t, y) = K((t, I)^{-1}x, y) = K(x, (t, I)y) = K(x, y + t),$$

or upon substituting $x \leftarrow x + t$, $K(x, y) = K(x + t, y + t)$. Choosing t to be the translation that takes y to 0, we find that

$$K(x, y) = K(x - y, 0) =: k(x - y)$$

defines a convolution kernel $k : \mathbf{R}^d \to \mathrm{Hom}(\mathbf{R}^{d_X}, \mathbf{R}^{d_Y})$. Now specialising Equation (2.7) by letting $R \in H$ and $x \in \mathbf{R}^d$ be arbitrary and $t, y = 0$, we obtain the condition $\pi_Y(R)k(R^{-1}x) = k(x)\pi_X(R)$, or upon substituting $x \leftarrow Rx$ and rearranging,

$$k(Rx) = \pi_Y(R)k(x)\pi_X(R^{-1}). \tag{2.8}$$

The above reasoning can be reversed to show that the condition in Equation (2.8) (for all $x \in \mathbf{R}^d, R \in H$) is sufficient to guarantee equivariance of Φ.

The condition in Equation (2.8) is a linear constraint that is fully specified before training. Hence, if a basis is computed for the convolution kernels satisfying Equation (2.8), a general equivariant linear operator can be learned by learning its parameters in that basis. Since the choices of H that we consider are all compact groups, any representation of H can be decomposed as a direct sum of irreducible representations of H (Theorem 5.2 in Folland (2015)). As a result of this, we can give the following procedure to compute a basis for the convolution kernels satisfying the equivariance condition in Equation (2.8) as soon as π_X and π_Y are specified:

- Decompose π_X and π_Y as direct sum of irreducible representations:

  $$\pi_X = Q_X \,\mathrm{diag}(\pi_X^1, \ldots, \pi_X^{k_X})Q_X^{-1}, \quad \pi_Y = Q_Y \,\mathrm{diag}(\pi_Y^1, \ldots, \pi_Y^{k_Y})Q_Y^{-1}.$$

  Here diag constructs a block diagonal matrix with the diagonal elements given by the arguments supplied to diag.

- For each i, j with $1 \leqslant i \leqslant k_X, 1 \leqslant j \leqslant k_Y$ find a basis for the convolution kernels $k_{i,j}$ satisfying the equivariance condition

  $$k_{i,j}(Rx) = \pi_Y^j(R)k_{i,j}(x)\pi_X^j(R^{-1})$$

  with the irreducible representations π_Y^j and π_X^i.

- Given expansions of the $k_{i,j}$, compute the overall equivariant convolution kernel k by

$$k = Q_{\mathcal{Y}} \cdot (k_{i,j})_{1 \leqslant i \leqslant k_{\mathcal{X}}, 1 \leqslant j \leqslant k_{\mathcal{Y}}} \cdot Q_{\mathcal{X}}^{-1}.$$

This procedure has been described in more detail in Weiler and Cesa (2019) and implemented in the corresponding software package for the groups $G = \mathbf{R}^2 \rtimes H$, where H can be any subgroup of $\mathrm{O}(2)$.

Since the equivariant convolutions described above are implemented using ordinary convolutions, little extra computational effort is required to use them compared to ordinary convolutions: during training, there is just an additional step of computing the basis expansion defining the equivariant convolution kernels (and backpropagating through it). When it is time to test the network, this step can be avoided by computing the basis expansion once and only saving the resulting convolution kernels, so that it is completely equivalent in terms of computational effort to using an ordinary CNN.

Example 2.3.2. To get a feeling for how the above procedure works in practice, let us consider an example that is relevant to the methods that we will describe in Section 2.4. Suppose that $H = \mathbf{Z}_4 = \{\mathrm{id}, r, r^2, r^3\}$ is the group of on-grid rotations, where

$$r = \begin{pmatrix} \cos(\pi/2) & -\sin(\pi/2) \\ \sin(\pi/2) & \cos(\pi/2) \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

Although it is neater to work over the complex field (since H is abelian, all complex representations of H are 1-dimensional), we will stick to the real approach described in Weiler and Cesa (2019). H has three irreducible irrepresentations:

$$\pi^1(r^n) = 1 \,(\text{the trivial representation}), \quad \pi^2(r^n) = r^n \quad \text{and} \quad \pi^3(r^n) = (-1)^n.$$

The inputs and outputs of the neural networks we use are scalar fields, i.e. they transform according the trivial representation π^1 under rotations. In addition, we will use intermediate features in the neural networks that transform according to the regular representation:

$$\pi^{\mathrm{reg}}(r^n) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}^n.$$

As is true in general, the regular representation decomposes as a direct sum of irreducible representations, each of which occurs with multiplicity 1:

$$\pi^{\text{reg}} = Q\,\text{diag}(\pi^1, \pi^2, \pi^3)Q^{-1}, \quad \text{with} \quad Q = \frac{1}{2}\begin{pmatrix} 1 & \sqrt{2} & 0 & 1 \\ 1 & 0 & \sqrt{2} & -1 \\ 1 & -\sqrt{2} & 0 & 1 \\ 1 & 0 & -\sqrt{2} & -1 \end{pmatrix}.$$

Following the above reasoning, we can reduce the equivariance condition in Equation (2.8) to the conditions

$$k_{i,j}(r^n x) = \pi^i(r^n)k(x)\pi^j(r^n). \tag{2.9}$$

These constraints are most easily solved by switching to polar coordinates and expanding the angular part of the kernel in Fourier series. The interested reader can refer to the appendices of Weiler and Cesa (2019) for the full computations, but the upshot is that each equivariance condition is equivalent to restricting the angular part to only consist of certain Fourier modes (and finitely many of them). For the radial part, we multiply by rings of Gaussian kernels centered at various radii. Finally, we sample the obtained functions on a discrete grid to obtain a basis of equivariant filters. Figure 2.2 shows an example of the result of this procedure for $i = 1, j = 2$, discretising on a grid of size 7×7. After discretisation, the equivariant convolution is performed simply using ordinary convolutions with these filters, padding the inputs with zeros to ensure that the outputs are of the same size.

Figure 2.2: An example of a discretised basis of equivariant filters on a grid of size 7×7. The filters are based on solving the reduced equivariance condition in Equation (2.9) for $i = 1, j = 2$. Each row corresponds to one of the components of the kernel $k_{1,2}$.

2.3.2 Equivariant nonlinearities

Although pointwise nonlinearities are translationally equivariant, some more care is needed when designing nonlinearities that satisfy the equivariance condition in Equation (2.5) with our choices of groups. Examining the form of the induced representations in our setting, as given in Equation (2.6), it is evident that for a pointwise nonlinearity $\phi : \mathbf{R} \to \mathbf{R}$ to be equivariant (in the sense that $\phi(\rho_{\mathcal{X}}(g)[f]) = \rho_{\mathcal{X}}(g)[\phi(f)]$, with ϕ applied pointwise) ϕ must commute with $\pi_{\mathcal{X}}(R)$ for every $R \in H$: with $g = (t, R)$ for $t \in \mathbf{R}^d, R \in H$ we have

$$\phi(\pi_{\mathcal{X}}(R) f(g^{-1}x)) = \phi(\rho_{\mathcal{X}}(g)[f])(x) = \rho_{\mathcal{X}}(g)[\phi(f)](x) = \pi_{\mathcal{X}}(h)\phi(f(g^{-1}x)).$$

This can be ensured if $\pi_{\mathcal{X}}$ is the regular representation of H, since in that case each $\pi_{\mathcal{X}}(h)$ is a permutation matrix, giving the following guideline:

Lemma 2.3.2. *Suppose that $G = \mathbf{R}^d \rtimes H$ with H a finite subgroup of $\mathrm{O}(d)$ and that $\phi : \mathbf{R} \to \mathbf{R}$ is a given function. If $\pi_{\mathcal{X}}$ is the regular representation of H, then $\Phi : \mathcal{X} \to \mathcal{X}$ is equivariant, where $\Phi(f)(x) = \phi(f(x))$.*

Another way to ensure that ϕ commutes with $\pi_{\mathcal{X}}$ is by choosing the trivial representation. Although the trivial representation may not be very interesting by itself, this gives rise to another form of nonlinearity called the norm nonlinearity. If $\pi_{\mathcal{X}}$ is a unitary representation, taking the pointwise norm satisfies an equivariance condition: with $g = (t, R)$ for $t \in \mathbf{R}^d, R \in H$

$$\|\rho_{\mathcal{X}}(g)[f](x)\| = \|\pi_{\mathcal{X}}(R) f(g^{-1}x)\| = \|f(g^{-1}x)\|.$$

The right-hand side transforms according to the trivial representation, so by the above comments we deduce that the nonlinearity $f \mapsto \phi(\|f\|)$ satisfies an equivariance condition of the same form. To obtain the norm nonlinearity, which maps features of a given type to features of the same type, we then form the map $\Phi : \mathcal{X} \to \mathcal{X}, f \mapsto f \cdot \phi(\|f\|)$: with $g = (t, R)$ for $t \in \mathbf{R}^d, R \in H$, we have

$$\begin{aligned}
\Phi(\rho_{\mathcal{X}}(g)[f])(x) &= \pi_{\mathcal{X}}(R) f(g^{-1}x) \cdot \phi(\|f(g^{-1}x)\|) \\
&= \pi_{\mathcal{X}}(R)\Big(f(g^{-1}x) \cdot \phi(\|f(g^{-1}x)\|)\Big) \\
&= \pi_{\mathcal{X}}(R)\Big(f \cdot \phi(\|f\|)\Big)(g^{-1}x) \\
&= \rho_{\mathcal{X}}(g)[\Phi(f)](x),
\end{aligned}$$

where we used the fact that $\phi(\|f(g^{-1}x)\|)$ is a scalar. This shows that the norm nonlinearity Φ is indeed equivariant:

Lemma 2.3.3. *Suppose that $\pi_{\mathcal{X}}$ is a unitary representation of H, and that $\phi : \mathbf{R} \to \mathbf{R}$ is a given function. Then the norm nonlinearity $\Phi : \mathcal{X} \to \mathcal{X}$ with $\Phi(f)[x] = f(x)\phi(\|f(x)\|)$ is equivariant.*

2.4 Reconstruction methods motivated by variational regularisation

We consider the inverse problem of estimating an image u from noisy measurements y. We will assume that knowledge of the measurement process is available in the form of the forward operator A, which maps an image to ideal, noiseless measurements, and generally there will be a reasonable idea of the process by which they are corrupted to give rise to the noisy measurements y. A tried and tested approach to solving inverse problems is the variational regularisation approach (Burger and Osher, 2004; Engl et al., 1996). In this approach, images are recovered from measurements by minimising a trade-off between the data fit and a penalty function encoding prior knowledge:

$$\hat{u} = \operatorname*{argmin}_u E_y(u) + J(u), \tag{2.10}$$

with E_y a data discrepancy functional penalising mismatch of the estimated image and the measurements and J the penalty function. Usually E_y will take the form $E_y(u) = d(A(u), y)$, where d is a measure of divergence chosen based on our knowledge of the noise process.

2.4.1 Equivariance in splitting methods

Generally, Problem (2.10) may be difficult to solve, and a lot of research has been done on methods to solve problems such as these. Iterative methods to solve it are often structured as splitting methods: the objective function is split into terms, and easier subproblems associated with each of these terms are solved in an alternating fashion to yield a solution to Problem (2.10) in the limit. A prototypical example of this is the proximal gradient method (also known as forward-backward splitting) (Bruck, 1977; Passty, 1979), which has become a standard tool for solving linear inverse problems, particularly in the form of the FISTA algorithm (Beck and Teboulle, 2009). In its basic form, the proximal gradient method performs the procedure described in Algorithm 4.

Recall here that the proximal operator (Moreau, 1962, 1963, 1965) prox_J is defined as follows:

Definition 2.4.1. Suppose that $\mathcal{X}$ is a Hilbert space and that $J : \mathcal{X} \to \mathbf{R} \cup \{+\infty\}$ is a lower semi-continuous convex proper functional. The proximal operator $\operatorname{prox}_J : \mathcal{X} \to \mathcal{X}$ is then defined as

$$\operatorname{prox}_J(u) = \operatorname*{argmin}_{u' \in \mathcal{X}} \frac{1}{2}\|u - u'\|^2 + J(u') \tag{2.11}$$

Algorithm 4 Proximal gradient method

inputs: measurements y, initial estimate u^0
$u \leftarrow u^0$
for $i \leftarrow 1, \ldots,$ `it` **do**
    $u \leftarrow \operatorname{prox}_{\tau^i J}(u - \tau^i \nabla E_y(u))$
end for
return u

Although this definition of proximal operators assumes that the functional J is convex, this assumption is more stringent than is necessary to ensure that an operator defined by Equation (2.11) is well-defined and single-valued. One can point for example to the classes of μ-semi-convex functionals (i.e. the set of J, such that $u \mapsto J(u) + \frac{\mu}{2}\|u\|^2$ is convex) on $\mathcal{X}$ for $0 < \mu < 1$, which include nonconvex functionals. In what follows, we will allow for such more general functionals by just asking that the proximal operator is well-defined and single-valued.

It is often reasonable to ask that the proximal operators $\operatorname{prox}_{\tau J}$ satisfy an equivariance property; if the corresponding regularisation functional J is invariant to a group symmetry, the proximal operator will be equivariant:

Proposition 2.4.1. *Suppose that $\mathcal{X}$ is a Hilbert space and ρ is a unitary representation of a group G on $\mathcal{X}$. If a functional $J : \mathcal{X} \to \mathbf{R} \cup \{+\infty\}$ is invariant, i.e. $J(\rho(g)f) = J(f)$, and has a well-defined single-valued proximal operator $\operatorname{prox}_J : \mathcal{X} \to \mathcal{X}$, then prox_J is equivariant, in the sense that*

$$\operatorname{prox}_J(\rho(g)f) = \rho(g)\operatorname{prox}_J(f)$$

for all $f \in \mathcal{X}$ and $g \in G$.

Proof. We have the following chain of equalities:

$$\begin{aligned}
\operatorname{prox}_J(\rho(g)f) &= \operatorname*{argmin}_h \frac{1}{2}\|\rho(g)f - h\|^2 + J(h) \\
&\overset{(a)}{=} \operatorname*{argmin}_h \frac{1}{2}\|\rho(g)(f - \rho(g^{-1})h)\|^2 + J(\rho(g^{-1})h) \\
&\overset{(b)}{=} \operatorname*{argmin}_h \frac{1}{2}\|f - \rho(g^{-1})h\|^2 + J(\rho(g^{-1})h) \\
&\overset{(c)}{=} \rho(g)[\operatorname*{argmin}_h \frac{1}{2}\|f - h\|^2 + J(h)] = \rho(g)\operatorname{prox}_J(f).
\end{aligned}$$

The three marked steps are justified as follows:

(a) J is assumed to be invariant w.r.t. ρ,

(b) The representation ρ is assumed to be unitary,

(c) $\rho(g)$ is invertible, and under the substitution $h \leftarrow \rho(g)h$, the minimiser transforms accordingly.

□

Example 2.4.1. As a prominent example of a regularisation functional satisfying the conditions of Proposition 2.4.1, consider the total variation functional (Rudin et al., 1992) on $L^2(\mathbf{R}^d)$

$$\mathrm{TV}(u) = \sup_{\phi \in C_c^\infty(\mathbf{R}^d;\mathbf{R}^d), \|\phi\|_\infty \leqslant 1} \int_{\mathbf{R}^d} u \operatorname{div} \phi,$$

with the group $G = \mathrm{SE}(d)$ and the scalar field representation $\rho(r)[f](x) = f(r^{-1}x)$. Since the Lebesgue measure is invariant to G and the set of vector fields $\{\phi \in C_c^\infty(\mathbf{R}^d;\mathbf{R}^d) | \|\phi\|_\infty \leqslant 1\}$ is closed under G, TV is invariant w.r.t. ρ. As a result of this, Proposition 2.4.1 tells us that $\mathrm{prox}_{\tau\,\mathrm{TV}}$ is equivariant w.r.t. ρ for any $\tau \geqslant 0$. Note that TV is not unique in satisfying these conditions; by a similar argument it can be shown, for example, that the higher order total generalised variation functionals (Bredies et al., 2010) share the same invariance property (and hence also that their proximal operators are equivariant).

Remark 2.4.1. The above example, and all other examples that we consider in this chapter, are concerned with the case where the image to be recovered is a scalar field. Note, however, that Proposition 2.4.1 is not limited to this type of field and that there are applications where it is natural to use more complicated representations ρ. A notable example is diffusion tensor MRI (Coulon et al., 2004) in which case the image to be estimated is a diffusion tensor field and ρ should be chosen as the appropriate tensor representation.

Equivariance of the reconstruction operator

It is worth thinking about whether it is sensible to ask that the overall reconstruction method is equivariant, and how this should be interpreted. Thinking of the reconstruction operator as a map from measurements y to images $\hat{u}$, it is hard to make sense of the statement that it is equivariant, since the measurement space generally does not share the symmetries of the image space (in the case where measurements may be incomplete). If we think instead of the reconstruction method as mapping a true image u to an estimated image $\hat{u}$ through (noiseless) measurements $y = A(u)$, we might ask that a symmetry transformation of u should correspond to the same symmetry transformation of $\hat{u}$. In the case of reconstruction by a

variational regularisation method as in Problem (2.10), this is too much to ask for even if the regularisation functional is invariant, since information in the (incomplete) measurements can appear or disappear under symmetry transformations of the true image. An example of this phenomenon when solving an inpainting problem is shown in Figure 2.3.

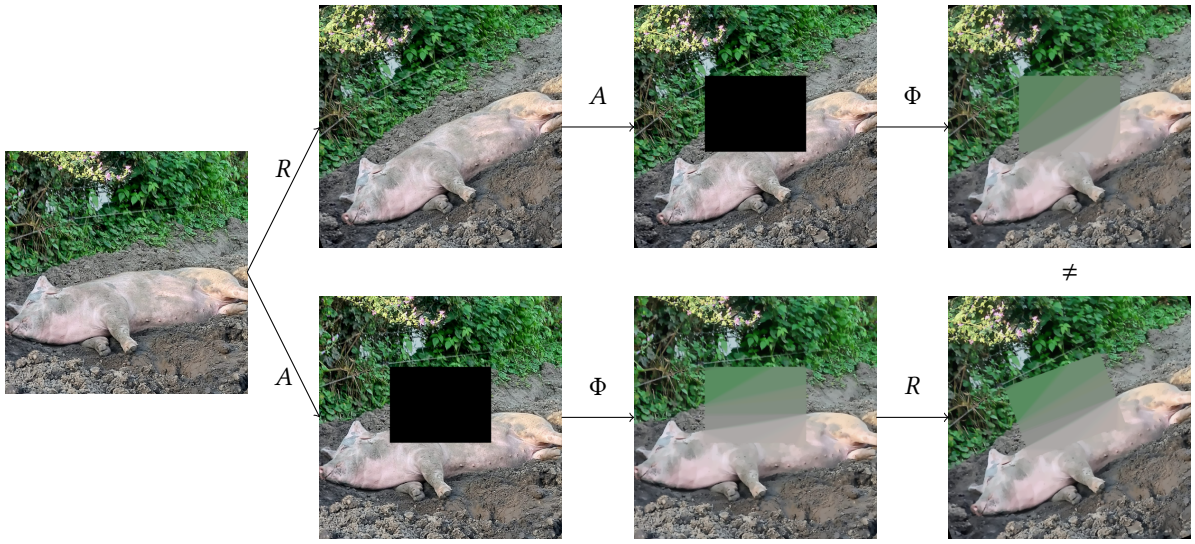

Figure 2.3: An example demonstrating the non-equivariance of a general variational regularisation approach to image reconstruction, even when the corresponding regularisation functional J (as in Problem (2.10)) is invariant. Here, A represents the application of an inpainting mask, R is an operator rotating the image by 20° and Φ is the solution map to Problem (2.10) with $E_y(u) = \|Au - y\|^2$ and $J(u) = \tau\,\mathrm{TV}(u)$.

2.4.2 Learned proximal gradient descent

A natural way to use knowledge of the forward model in a neural network approach to image reconstruction is in the form of unrolled iterative methods (Adler and Öktem, 2017; Putzky and Welling, 2017). Starting from an iterative method to solve Problem (2.10), the method is truncated to a fixed number of iterations and some of the steps in the truncated algorithm are replaced by learnable parts. As noted in the previous section, the proximal gradient method in Algorithm 4 can be applied to a variational regularisation problem such as Problem (2.10). Motivated by this and the unrolled iterative method approach, we can study learned proximal gradient descent as in Algorithm 5 (where the variable s can be used as a memory state, as is common in accelerated versions of the proximal gradient method (Beck and Teboulle, 2009)):

Algorithm 5 Learned proximal gradient method

inputs: measurements y, initial estimate u^0
$u \leftarrow u^0, s \leftarrow 0$
for $i \leftarrow 1, \ldots,$ `it` **do**
  $(u, s) \leftarrow \widehat{\text{prox}}_i(u, s, \nabla E_y(u))$
end for
return $\Phi(y) := u$

Here $\widehat{\text{prox}}_i$ are neural networks, the architectures of which are chosen to model proximal operators. In this work, we choose $\widehat{\text{prox}}_i$ to be defined as

$$\widehat{\text{prox}}_i = K_{\text{project},i} \circ (\text{id} + \phi \circ K_{\text{intermediate},i}) \circ K_{\text{lift},i}, \tag{2.12}$$

where each of the $K_{\text{project},i}, K_{\text{intermediate},i}$ and $K_{\text{lift},i}$ are learnable affine operators (given by a convolution operation followed by adding a bias term) and ϕ is an appropriate nonlinear function. We can appeal to Proposition 2.4.1 and model $\widehat{\text{prox}}_i$ as translationally equivariant (we will call the corresponding reconstruction method the ordinary method in what follows) or as roto-translationally equivariant (we will call the corresponding reconstruction method the equivariant method in what follows). Figure 2.4 gives a schematic illustration of the inputs and outputs of the learned proximal operators.

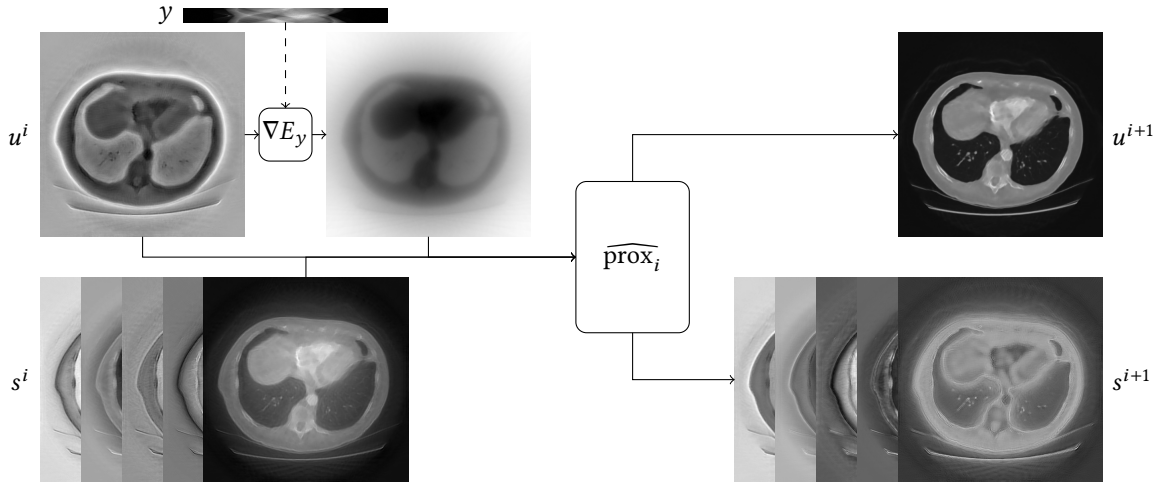

Figure 2.4: A schematic illustration of a single iteration of the learned proximal gradient method, Algorithm 5, for a CT reconstruction problem. The choice of E_y is described in Section 2.5.1. Knowledge of the forward model is incorporated into the reconstruction through ∇E_y, which is not an equivariant operator in general. Motivated by Proposition 2.4.1, we know that $\widehat{\text{prox}}_i$ is naturally modelled as an equivariant operator.

Recall that we consider groups of the form $G = \mathbf{R}^d \rtimes H$ for subgroups H of $\mathrm{O}(d)$ in this chapter. Since we apply the learned equivariant method to reconstruct scalar-valued images, the input and output types of each $\widehat{\mathrm{prox}}_i$ should correspond to features carrying the trivial representation of H. For the equivariant method, $K_{\mathrm{lift},i}$ are equivariant convolutions from a small number (2 + the number of channels used for the memory state) of input channels with the trivial representation of H to a larger number of intermediate channels with the regular representation of H, if H is a finite group, or various irreducible representations of H, if H is a continuous group. $K_{\mathrm{intermediate},i}$ are chosen as equivariant convolutions mapping the output channels of $K_{\mathrm{lift},i}$ to a set of channels of the same type. Finally, $K_{\mathrm{project},i}$ are chosen as equivariant convolutions that map the output channels of $K_{\mathrm{intermediate},i}$ to a small number (1 + the number of channels used for the memory states) of output channels with the trivial representation of H. For the implementation of the equivariant convolutions, recall the procedure described at the end of Section 2.3.1.

For the ordinary method, $K_{\mathrm{lift},i}$ are ordinary convolutions mapping a small number (equal to that of the equivariant method) of input channels to a larger number of intermediate channels, $K_{\mathrm{intermediate},i}$ are ordinary convolutions mapping the output channels of $K_{\mathrm{lift},i}$ to a set of channels of the same type, and $K_{\mathrm{project},i}$ are ordinary convolutions mapping the many output channels of $K_{\mathrm{intermediate},i}$ to a small number (equal to that of the equivariant method) of output channels.

Since the implementations of the equivariant convolutions are ultimately based on ordinary convolutions, a natural comparison can be made between the equivariant and ordinary method by matching the widths of the underlying ordinary convolutions. When the methods are compared in this way, they should take comparable computational effort to use and the ordinary method is a superset of the equivariant method in the sense that the parameters of the ordinary method can be chosen to reproduce the action of the equivariant method.

Remark 2.4.2. Both in the case of Algorithm 4 and Algorithm 5, we require access to the gradient ∇E_y, where E_y is a data discrepancy functional. In our case, E always takes the form $E_y(u) = d(A(u), y)$ where A is the forward operator and d is a measure of divergence. As a result of this E_y can be differentiated by the chain rule as long as we have access to the gradient of d and can compute vector-Jacobian products of A. If the forward operator A is linear, its vector-Jacobian products are given just by the action of the adjoint of A.

2.5 Experiments

In this section, we demonstrate that roto-translationally equivariant operations can be incorporated into a learned iterative reconstruction method such as Algorithm 5 to obtain higher quality reconstructions than those obtained using comparable reconstruction methods that only use translationally equivariant operations. We consider two different inverse problems: a subsampled MRI problem and a low-dose CT problem. The code used to produce the experimental results shown is available at `https://github.com/fsherry/equivariant_image_recon`.

2.5.1 Datasets

LIDC-IDRI dataset

We use a selection of chest CT images of size 512×512 from the LIDC-IDRI dataset (Armato III et al., 2015, 2011) for our CT experiments. We use a combination of L^1 norm and the TV functional as a simple way to screen out low-quality images. The details of this procedure can be found in the code repository associated with this chapter. The set is split into 5000 images that can be used for training, 200 images that can be used for validation and 1000 images that can be used for testing. For the experiments using this dataset, we use the ASTRA toolbox (Aarle et al., 2016; Palenstijn et al., 2011; van Aarle et al., 2015) to simulate a parallel beam ray transform $\mathcal{R}$ with 50 uniformly spaced views at angles between 0 and π. We simulate the measurements y as post-log data in a low-dose setting:

$$y = -\frac{1}{\mu}\log\left(\max\left\{\frac{n}{N_{\text{in}}}, \eta\right\}\right), \quad \text{where} \quad n \sim \text{Pois}(N_{\text{in}}\exp(-\mu\mathcal{R}(u))).$$

Here $N_{\text{in}} = 10000$ is the average number of photons per detector pixel (without attenuation), μ is a base attenuation coefficient connecting the volume geometry and attenuation strength, and η is a small constant to ensure that the argument of the logarithm is strictly positive, chosen as $\eta = 10^{-8}$ in our experiments. Figure 2.5 shows some examples of the ground truth images and filtered backprojection reconstructions from the corresponding simulated measurements. In these experiments, we will define the data discrepancy functional E_y as

$$E_y(u) = \frac{1}{2}\|\mathcal{R}u - y\|_2^2.$$

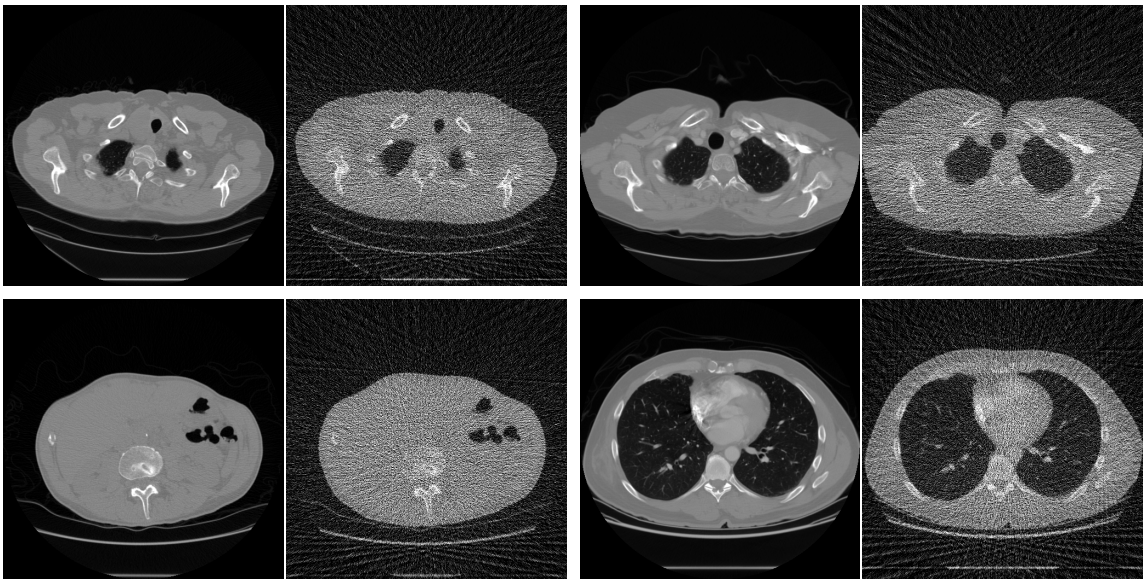

Figure 2.5: Four samples of the images that were used to train the reconstruction operators in the CT experiments, and the results of applying filtered backprojection (FBP) to the corresponding simulated sinograms. The images are clipped between upper and lower attenuation coefficient limits of −1024 HU and 1023 HU.

FastMRI

We use a selection of axial T1-weighted brain images of size 320×320 from the FastMRI dataset (Knoll et al., 2020; Zbontar et al., 2019) for our MRI experiments. As in Section 2.5.1, we screen the images to remove as many low-quality images as possible, and we split the remaining images into training, validation and test sets in the same way as we did previously. For the experiments using this dataset, we simulate the measurements using a discrete Fourier transform $\mathcal{F}$ and a variable density Cartesian line sampling pattern $\mathcal{S}$ (simulated using the software package associated with the work in Lustig et al. (2007b) and shown in Figure 2.6):

$$y = \mathcal{S}\mathcal{F}u + \varepsilon,$$

where ε is complex-valued white Gaussian noise. In this setting, a complex-valued image is modelled as a real image with two channels, one for the real part and the other for the imaginary part. The corresponding data discrepancy functional (E_y in Equation (2.10)) will be defined as

$$E_y(u) = \frac{1}{2}\|\mathcal{S}\mathcal{F}u - y\|_2^2.$$

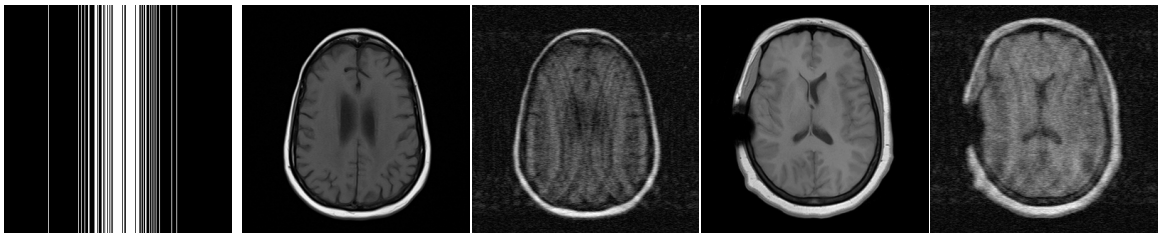

Figure 2.6: The sampling mask $\mathcal{S}$ used in the MRI experiments, sampling 20.3% of k-space, and two samples of the images that were used to train the reconstruction operators in the MRI experiments, and the zero-filling reconstructions from the corresponding simulated k-space measurements.

2.5.2 Experimental setup

Learning framework

Although it is also possible to learn the parameters of the reconstruction methods in Algorithm 5 in an unsupervised learning setting, all experiments that we consider in this chapter can be classified as supervised learning experiments: given a finite training set $\{(u_i^*, y_i)\}_{i=1}^N$ of ground truth images u_i and corresponding noisy measurements y_i, we choose the parameters of Φ in Algorithm 5 by solving the empirical risk minimisation problem

$$\min_{\Phi} \frac{1}{N} \sum_{i=1}^{N} \|u_i^* - \Phi(y_i)\|_2^2.$$

Architectures and initialisations of the reconstruction networks

We use the reconstruction networks defined in Section 2.4.2, referring to the architecture described there with roto-translationally equivariant components as the equivariant method and referring to the architecture with translationally equivariant components as the ordinary method. To ensure fair comparisons between the various methods that we compare, we fix as many as possible of the aspects of the methods that are not relevant to the point investigated in the experiments. To this end, every learned proximal gradient method has a depth of `it` $= 8$ iterations. Both for the CT and MRI experiment, the images being recovered are two-dimensional, so we use equivariant convolutions with respect to groups of the form $\mathbf{R}^2 \rtimes \mathbf{Z}_m$. Since the equivariant convolutions are implemented using ordinary convolutions, it is natural and straightforward to compare methods with the same width. The width of each network is the same (feature vectors that transform according to the regular representation take up $|H|$

"ordinary" channels, and we fix the size of the product $|H| \cdot n_{\text{channels}} = 96$ where n_{channels} is the number of such feature vectors in the intermediate part of $\widehat{\text{prox}}_i$ in Equation (2.12)). All convolution filters used are of size 3×3. We choose the initial reconstruction $u^0 = 0$ and use a memory variable s of five scalar channels wide in the learned proximal gradient method (Algorithm 5).

Furthermore we ensure that the initialisation of both types of methods are comparable. Referring back to Equation (2.12), we choose to initialise $K_{\text{intermediate},i}$ equal to zero and let $K_{\text{project},i}$ and $K_{\text{lift},i}$ be randomly initialised using the He initialisation method (He et al., 2015), as implemented in PyTorch (Paszke et al., 2019) for ordinary convolutions and generalised to equivariant convolutions in Weiler et al. (2018b) and implemented in the software package `https://github.com/QUVA-Lab/e2cnn` (Weiler and Cesa, 2019). For the practical implementation of the exact methods studied, the reader is advised to consult the code at `https://github.com/fsherry/equivariant_image_recon`.

Hyperparameters of the equivariant methods

In addition to the usual parameters of a convolutional neural network, the learned equivariant reconstruction methods have additional parameters related to the choice of the symmetry group and which of its representations to use. In this chapter, we have chosen to work with groups of the form $\mathbf{R}^2 \rtimes \mathbf{Z}_m$, so a choice needs to be made which $m \in \mathbf{N}$ to consider.

In Figure 2.7, we see the result of training and validating learned equivariant reconstruction methods on the CT reconstruction problem, with various orders m of the group $H = \mathbf{Z}_m$. Each of the learned methods is trained on the same training set consisting of 100 images. The violin plots used give kernel density estimates of the distributions of the performance measures; for each one, we have omitted the top and bottom 5% of values so as not to be misled by outliers. Evidently, in this case, the groups of on-grid rotations significantly outperform the other choices, with $m = 4$ giving the best performance. Based on this result, all further experiments with the equivariant methods will use the group $H = \mathbf{Z}_4$.

Training details

For both the equivariant and ordinary reconstruction methods, we train the methods using the Adam optimisation algorithm (Kingma and Ba, 2017) with learning rate 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$. We use minibatches of size 1 and perform a total of 10^5 iterations of the Adam algorithm to train each method, so that we perform the same total number of iterations for each training set, regardless of its size. Since we have chosen to use the finite group

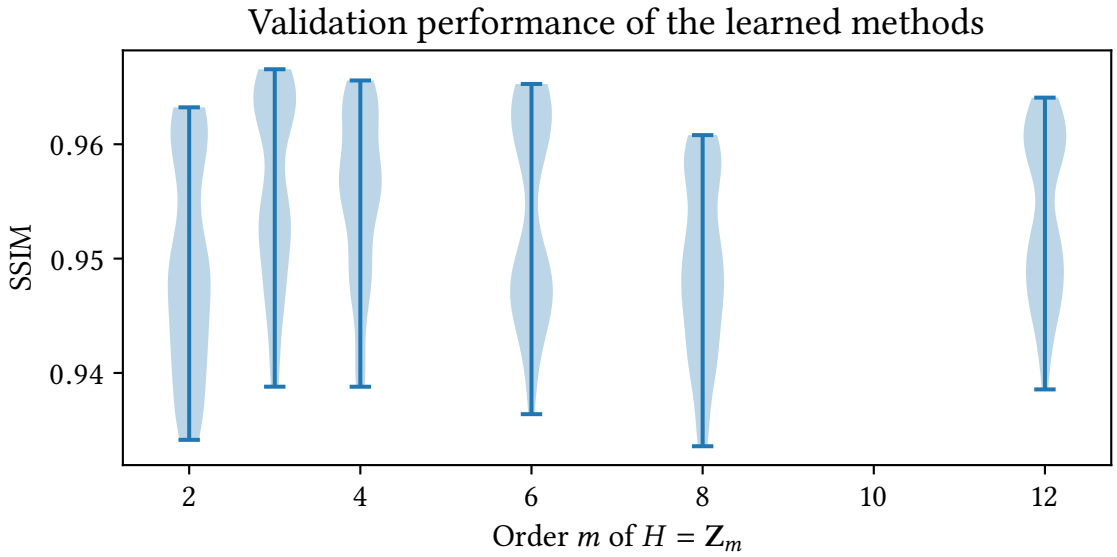

Figure 2.7: The reconstruction quality, as measured on a validation set, of learned proximal gradient methods trained on the CT reconstruction problem with varying orders of the group H. Note that when H is chosen to represent on-grid rotations (i.e. $m = 2$ or $m = 4$), the performance is significantly better than for any of the other choices of H.

approach, with intermediate fields transforming according to their regular representation, we can use a pointwise nonlinearity for both the equivariant and ordinary reconstruction methods. In all experiments, we use the leaky ReLU function as the nonlinearity (ϕ in Equation (2.12)), applied pointwise:

$$\phi(x) = \begin{cases} x & \text{if} \quad x > 0, \\ 0.01x & \text{else.} \end{cases}$$

Each training run is performed on a computer with an Intel Xeon Gold 6140 CPU and a NVIDIA Tesla P100 GPU. Training the equivariant methods requires slightly more computational effort than the ordinary methods: to begin with, given the specification of the architecture, bases need to be computed for the equivariant convolution kernels (this takes negligible effort compared to the effort expended in training). Besides this, each training iteration requires the computation of the convolutional filter from its parameters and the basis functions and the backpropagation through this basis expansion. To give an example of the extra computational effort required, we have timed 100 training iterations for comparable equivariant and ordinary methods for the MRI reconstruction problem: this took 35.5 seconds for the ordinary method and 41.9 seconds for the equivariant method, an increase of 18%. These times correspond to

a total training time of 9.9 hours and 11.6 hours for the equivariant and ordinary methods respectively. Note that at test time, however, the ordinary and equivariant methods can be computed with the same effort.

Performance measures

We will evaluate the reconstruction performance on regions of interest (the lung areas for the CT images and the general foreground region for the MRI images). Whereas the PSNR can immediately be applied to arbitrarily shaped signals (since the various locations in the signals do not interact), the SSIM in principle requires the input images to be regularly sampled to make sense of the subwindow statistics computed on windows of size $w \times w$ (see Appendix 3.4 for its definition). One way in which the SSIM can be reasonably computed on segmented data is as follows: Note that the subwindow SSIMs that are needed in the computation of the full SSIM define an image, the so-called SSIM map. If the input images are first padded on each side by $\lfloor w/2 \rfloor$ pixels (for example by reflection padding, as is done in the scikit-image implementation), the SSIM map computed from them will be of the same size as the original input images and will be aligned with them. The ordinary SSIM is computed by taking the average of such an SSIM map, so given a segmentation mask we can compute a segmented SSIM by instead taking the average of the values of the SSIM map over points that are inside the mask.

2.5.3 CT experiment: varying the size of the training set

In this experiment, we study the effect of varying the size of the training set on the performance of the equivariant and ordinary methods. We consider a range of training set sizes, as shown in Figure 2.8, and test the learned reconstruction methods on images that were not seen during training time, both in the same orientation and randomly rotated images. In medical applications, one tends to be particularly interested in the lung regions of the chest CT images. Although the methods have not been trained with this specifically in mind, in this section we will consider their performance on the lung regions. For this purpose, we use an automatic lung CT segmentation tool from Hofmanninger et al. (2020) to select the regions of interest. As can be seen in Figure 2.9, the equivariant method does a better job at reconstructing the lung regions than the ordinary method when trained on smaller training sets, but does slightly worse with larger training sets. This can be explained by the fact that the equivariant method is subsumed by the ordinary method (recall that the equivariant method can be replicated by appropriately setting the weights of the ordinary method, but the converse does not hold).

The violin plots displayed have the same interpretation as those shown in Figure 2.7 and described in Section 2.5.2. We see a slight deviation from a monotonic relationship between the training set size and reconstruction quality that would usually be expected. Small random variations in the test performance can be explained by various nondeterministic aspects of the training procedure: we use random initialisations of the network weights, the learning problem is nonconvex and there is randomness in how the examples are sampled during training. From this comparison, we see that the equivariant method is able to take better advantage of smaller training sets than the ordinary method. Furthermore, we see that the performance of the equivariant method does not suffer much when testing on images in unseen orientations, whereas the performance of the ordinary method drops significantly when testing on rotated images. Figure 2.9 shows some examples of test reconstructions made with the methods learned on a training set of size $N = 50$. In these reconstructions, it can be seen that the equivariant method does better at removing streaking artefacts than the ordinary method.

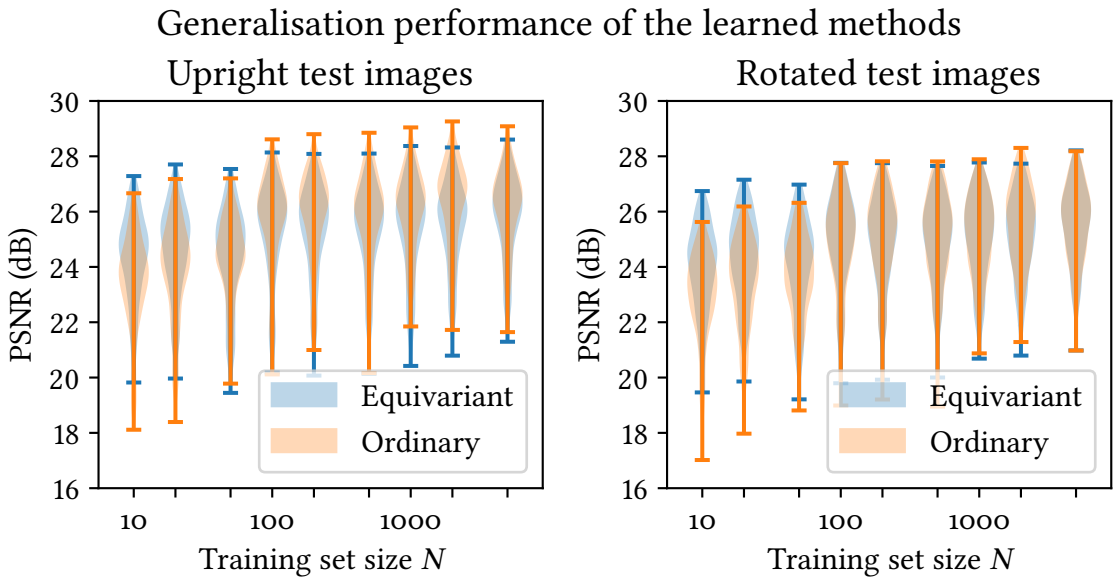

Figure 2.8: A comparison of the performance of equivariant and ordinary learned proximal gradient methods trained on training sets of various sizes for the CT reconstruction problem. The methods are tested on images that have not been seen during training time, both in the same orientations as were observed during training ("Upright test images") and rotated at random angles ("Rotated test images"). The performance is evaluated on the lung regions only.

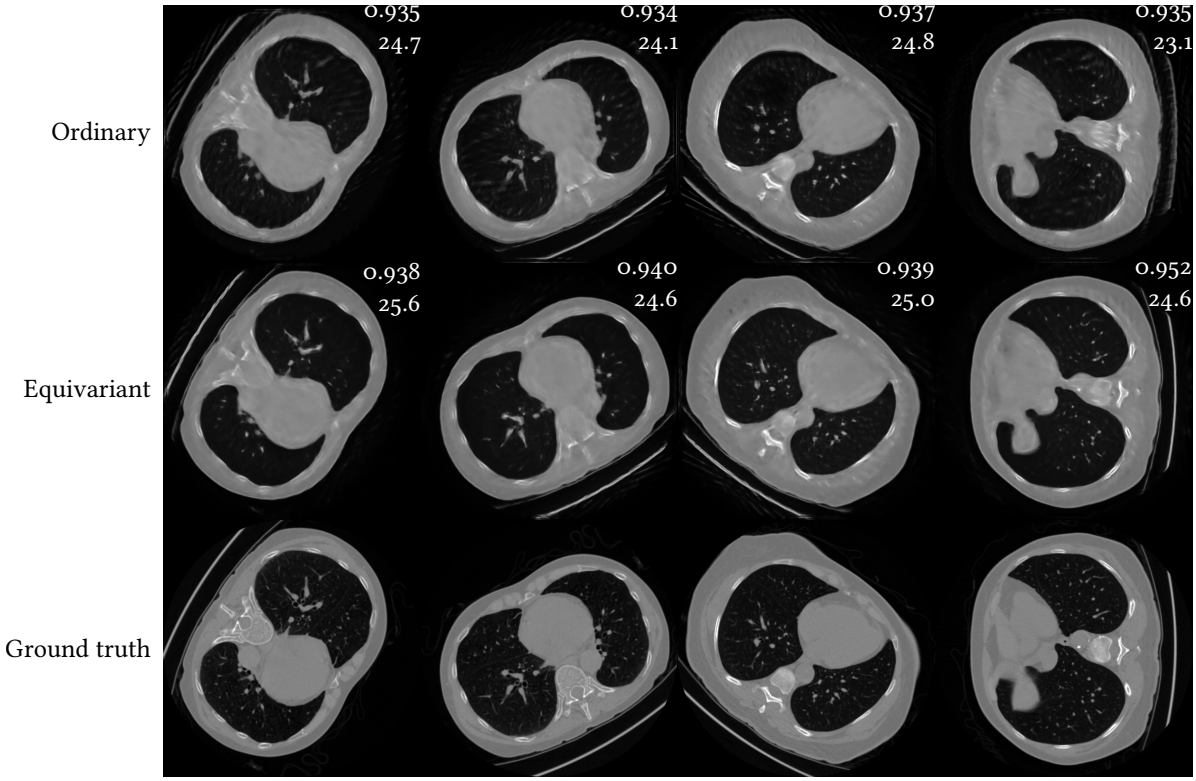

Figure 2.9: A random selection of test images corresponding to the plots shown in Figure 2.8, with a training set of size $N = 50$. On each reconstruction, the top number is its SSIM and the bottom number is its PSNR w.r.t. the ground truth, with both performance measures computed on the lung regions only. The images are clipped between upper and lower attenuation coefficient limits of -1024 HU and 1023 HU.

2.5.4 MRI experiment: varying the size of the training set

This experiment is similar to the experiment in Section 2.5.3, but concerns the MRI reconstruction problem. A notable difference with the CT reconstruction problem is that, as a result of the Cartesian line sampling pattern, the forward operator is now less compatible with rotational symmetry. Regardless of this, we have seen in Section 2.4 that it is still sensible in this context to use equivariant neural networks in a method motivated by a splitting optimisation method. As in section 2.5.3, we evaluate the performance of the learned methods on regions of interest: in this case we use the foreground of the images, which we isolate by thresholding the ground truth images, followed by taking the convex hull of the result. The performance differential between the equivariant and ordinary methods is more subtle than in the CT reconstruction problems. An explanation for this can be found in the fact that the MRI reconstruction problem is, in a certain sense, easier than the CT reconstruction problem:

the nonzero singular values of the MRI forward operator are constant, while those of the CT forward operator decay, complicating the inversion. Remarkably, it is observed that both methods perform better on rotated images than they do on upright images. This is an artefact of how the rotated images are created: rotated images are generated from the upright images by performing a rotation operation which necessarily includes an interpolation step. As a result of this, some of the high frequency details disappear after rotating, resulting in an easier reconstruction problem. Appendix 2.A goes into more detail about this effect. In Figure 2.10, we see that the equivariant method can again take better advantage of smaller training sets and is more robust to images dissimilar to those seen in training. Figure 2.11 shows examples of reconstructions made with the methods learned on a training set of size $N = 100$.

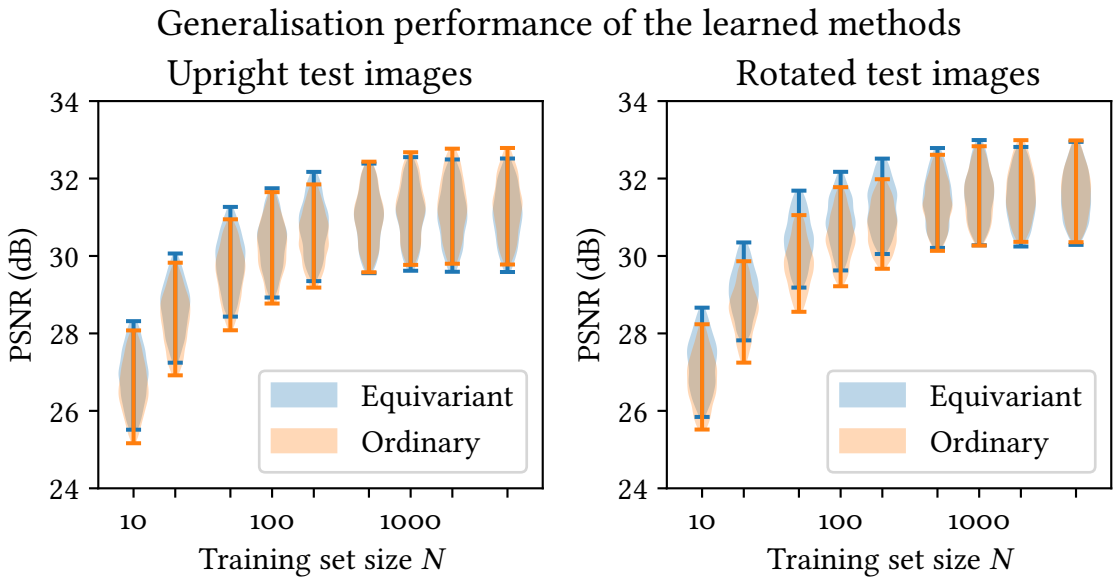

Figure 2.10: A comparison of the performance of equivariant and ordinary learned proximal gradient methods trained on training sets of various sizes for the MRI reconstruction problem. The methods are tested on images that have not been seen during training time and that have been rotated at random angles. The performance is evaluated on the foreground regions only.

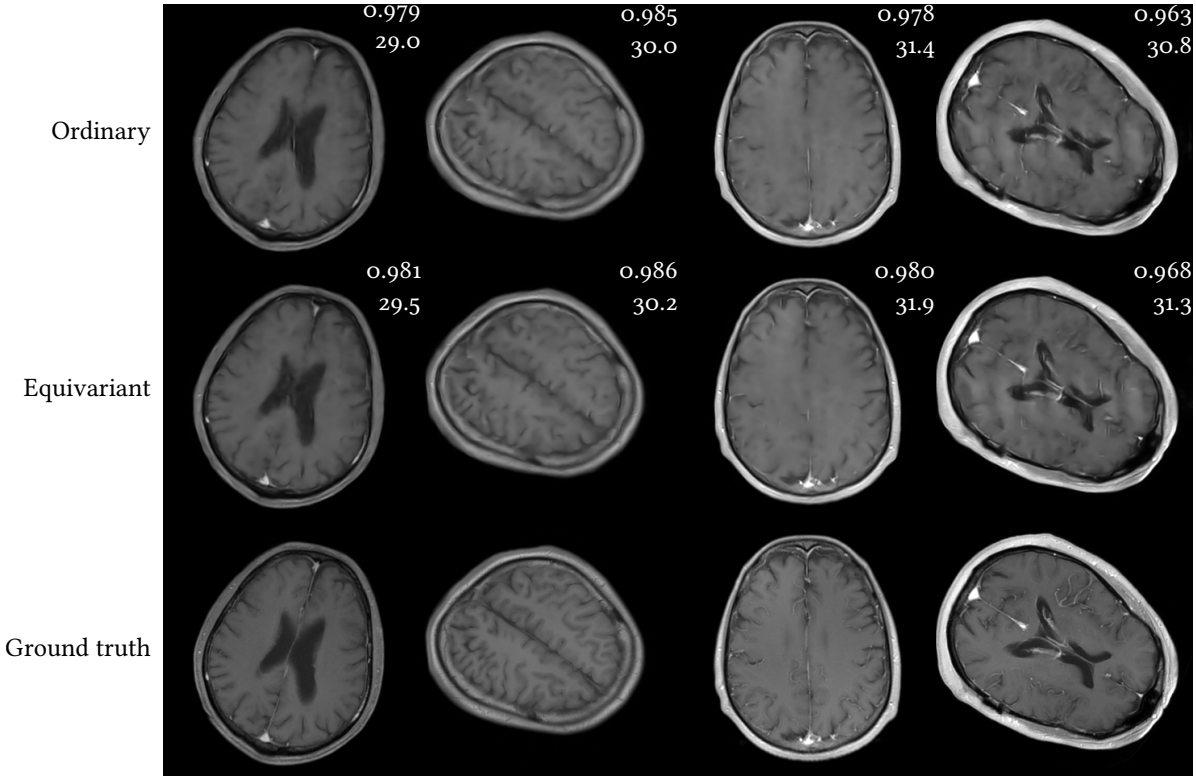

Figure 2.11: A random selection of test images corresponding to the plots shown in Figure 2.10, with a training set of size $N = 100$. On each reconstruction, the top number is its SSIM and the bottom number is its PSNR w.r.t. the ground truth, with both performance measures computed on the foreground regions only.

2.6 Conclusions and discussion

In this work, we have shown that equivariant neural networks can be naturally incorporated into learnable reconstruction methods for inverse problems. Doing so requires little extra effort and results in higher quality reconstructions when compared to similar methods that use ordinary convolutional neural networks. The main difference of this approach compared to existing approaches is that we model proximal operators in a learned reconstruction method as roto-translationally equivariant rather than just translationally equivariant, as is usually the case. Building the extra symmetries into the learned reconstruction method has the effect of lowering the method's sample complexity. Using roto-translationally equivariant neural networks as opposed to ordinary convolutional neural networks results in better performance when trained on smaller training sets and more robustness to rotations.

Let us now discuss some of the limitations of the proposed approach and potential improvements to be considered in future work.

As we saw in Section 2.5.3 and Section 2.5.4, the equivariant method outperforms the ordinary method for small training sets, but is slightly outperformed by the ordinary method for large training sets. This is a result of the equivariant method being a subset of the ordinary method. The equivariant method can be made more expressive by using a larger number of intermediate channels, but this comes at the expense of increased computational cost.

In Section 2.5.2, we saw that that the learned methods perform best when the group H is chosen to be a group of on-grid rotations. In theory, one would expect better performance with a larger number of rotations, but in practice there is the issue of how the equivariant kernels are discretised. Indeed, when solving the constraint for equivariance in Equation (2.8), the allowed kernels turn out to be circular harmonics multiplied by an arbitrary radial profile, and in practice we discretise these functions on 3×3 filters. An opportunity for future work on the use of equivariant neural networks can be found in how the combination of group and discretisation should be optimised.

All of the experiments shown in this work have dealt with two-dimensional images, but the methods described here can be applied equally well to three-dimensional images, as long as the two-dimensional equivariant convolutions are replaced by their three-dimensional counterparts. The representation theory of SO(3) is significantly more complicated than that of SO(2) (notably SO(2) is abelian but SO(3) is not), but it is similarly possible to design roto-translationally equivariant convolutions in three dimensions (Weiler et al., 2018a). One potential application is mentioned in Remark 2.4.1: in diffusion tensor MRI, the domain is

three-dimensional, with the additional challenge that the image that is to be recovered is a tensor field rather than a scalar field.

In the experiments that we demonstrated in this work, we focused on a single type of learned reconstruction operator, the learned proximal gradient method. In fact, the framework that we describe is not limited to this form of reconstruction algorithm. As an example of another type of learned reconstruction operator, consider the learned primal-dual method of Adler and Öktem (2018). A small corollary to Proposition 2.4.1 is that, when J is invariant and the Fenchel conjugate J^* is well-defined, $\operatorname{prox}_{J^*}$ will be equivariant in the same way that prox_J is. As a result, assuming reasonable invariance properties of a data discrepancy term, a learned primal-dual method can be considered where both the primal and dual proximal operators are modelled as appropriate equivariant neural networks.

Recall that the learned iterative reconstruction methods are modelled on the application of an iterative optimisation method to a variational regularisation problem. Let us consider what happens if we pursue this analogy further: we repeatedly apply a learned proximal gradient method of Algorithm 5, not just restricting to the number of iterations used to train it and record the progress through the learned proximal blocks as $\hat{u}^k_{\text{learned iterative}}$. To compare this to a conventional method, we consider the method we used in Chapter 1: we solve a variational regularisation problem with objective function as in Equation (1.3) using the Algorithm 1. We use a TV-style regularisation functional and tune the regularisation parameter, and denote the iterates by $\hat{u}^k_{\text{PDHG}}$ and the final result by $\hat{u}_{\text{PDHG}}$. Figure 2.12 shows the result of this comparison on a test image. As expected, the variational regularisation method shows a steady convergence behaviour. Much more striking is the result for the learned iterative reconstruction method: the intermediate iterates are not generally useful, there are periodic spikes at $k \equiv 0 \bmod \texttt{it}$ (where `it` is the number of iterations of the learned iterative reconstruction method), and in fact the iterates $\hat{u}^k_{\text{PDHG}}$ diverge as $k \to \infty$. On the other hand, we also see in Figure 2.12 that $\hat{u}^{\texttt{it}}_{\text{learned iterative}}$ (the iterate that is optimised during training time) is significantly better than the final result $\hat{u}_{\text{PDHG}}$ of the variational regularisation method. In conclusion, the learned iterative reconstruction method allows for computationally cheap, excellent quality, reconstructions, but these advantages come at the cost of moving away from variational regularisation methods and introducing a certain fragility as shown in the foregoing example. To take a step back in this direction again, we may impose that the learned "proximal operators" do not vary across iterations and that they have a certain stability to prevent the divergence shown in Figure 2.12. In the next chapter, we will study a way in which such stability, in the form of nonexpansiveness, can be built into neural network denoisers.

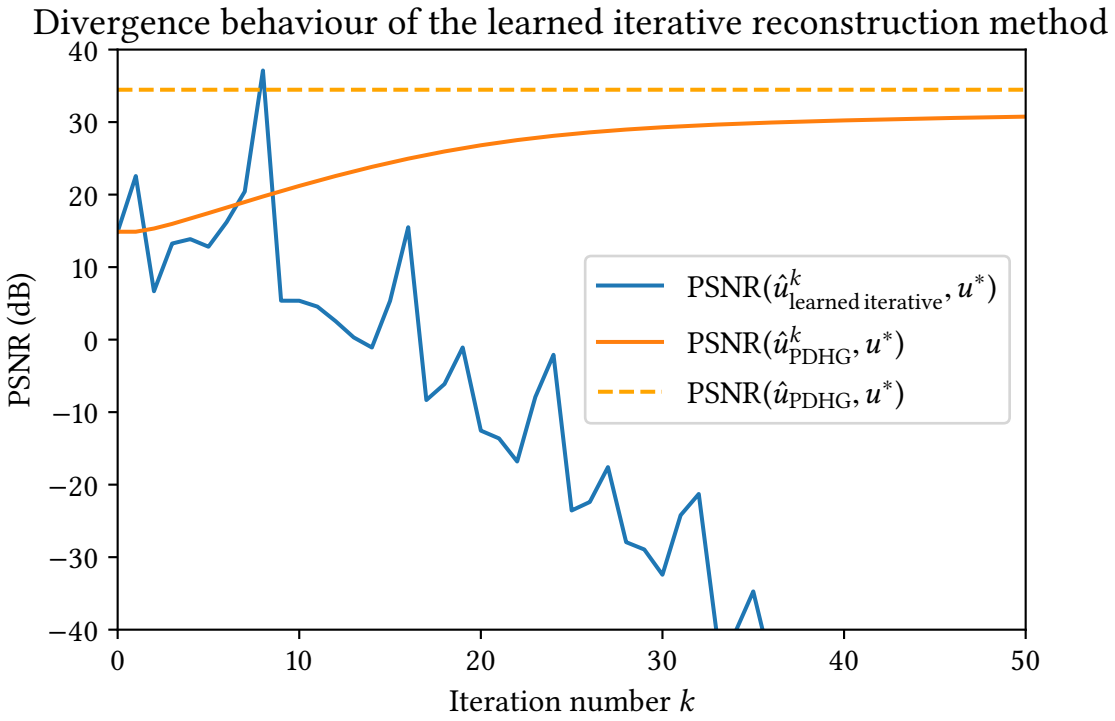

Figure 2.12: Divergence of a learned iterative reconstruction method when applied repeatedly, as compared to the convergence of a variational regularisation method. Notably, the iterate $\hat{u}^{\mathtt{it}}_{\text{learned iterative}}$ of the learned iterative reconstruction method is of excellent quality, but the other iterates are not.

Appendix 2.A The blurring effect of the rotation operation on discretised images

In Section 2.5.4, we made the remarkable observation that the learned reconstruction methods perform better for the MRI problem on rotated images than on upright images similar to those on which they were trained. It was mentioned there that this is an artefact of the way in which rotated images are created. As a simple test of this explanation, consider the comparison of the performance on the unaltered upright images and the performance on upright images that have been randomly rotated and then rotated back to be upright. If the hypothesised explanation for the difference in performance is correct, we would expect the methods to perform better on the images that have been rotated and rotated back than on the unaltered images. Figure 2.13 shows the result of doing this comparison, confirming that the MRI problem is significantly easier to solve for the learned reconstruction methods after the images have undergone the blurring effect of the rotation operation. Figure 2.14 shows the same comparison repeated for the CT problem. In this case the effect is still visible, but it is considerably weaker, which explains why it was not observed in Section 2.5.3.

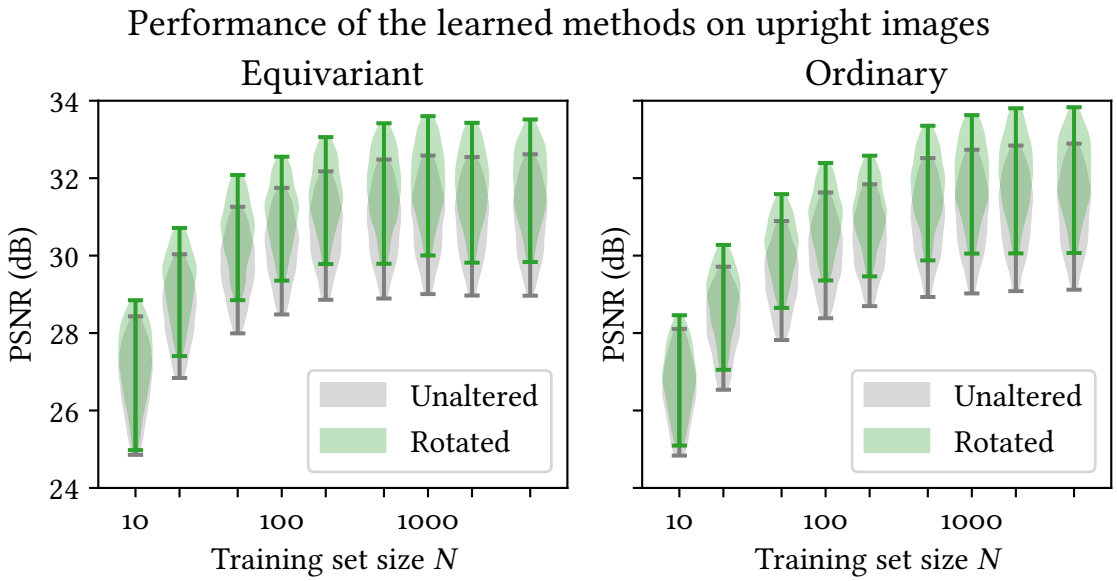

Figure 2.13: A comparison of the performance of the learned reconstruction methods on two types of upright images for the MRI problem: the original images ("Unaltered") and otherwise identical images that have been rotated and rotated back ("Rotated").

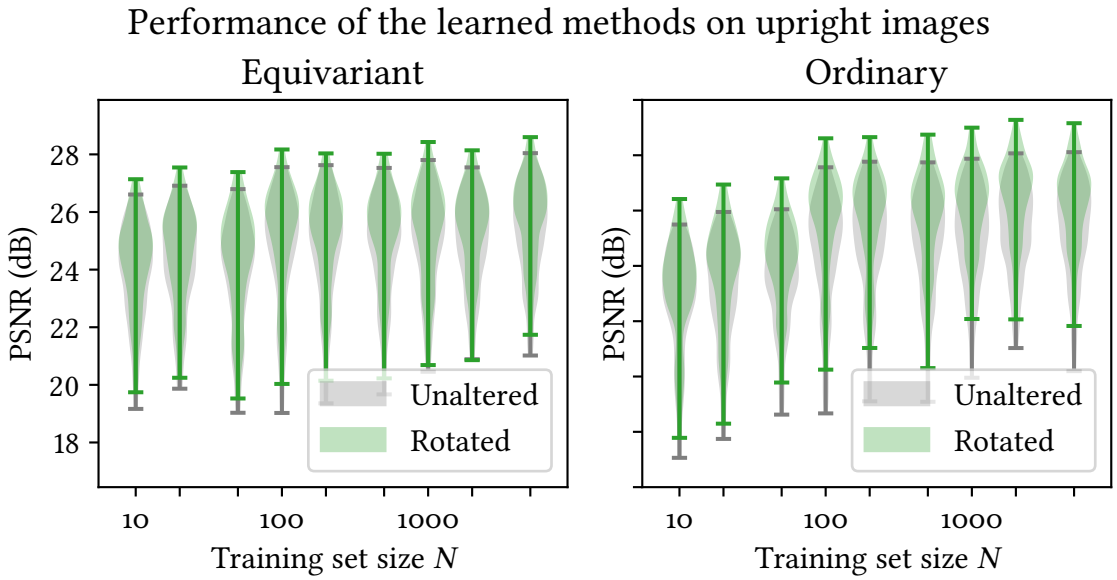

Figure 2.14: A comparison of the performance of the learned reconstruction methods on two types of upright images for the CT problem: the original images and otherwise identical images that have been rotated and rotated back.

Chapter 3

Nonexpansive neural networks inspired by ODEs and convex analysis

3.1 Introduction

As we saw in Figure 2.12 in the previous chapter, learned iterative reconstruction methods for inverse problems can give excellent reconstructions at a small fraction of the cost of iterative optimisation methods for variational regularisation problems, but do not preserve the convergence behaviour of the optimisation methods that they are modelled on. This can be seen as a disadvantage in a number of ways:

- Once a learned iterative reconstruction method has been trained, it is inflexible in the amount of computational effort required to make a reconstruction. In particular, unlike with most iterative optimisation methods for variational regularisation problems, we can not generally choose to make a different trade-off between computational effort and reconstruction quality than that decided during training time,
- Since we observe divergence when the learned iterative reconstruction method is repeatedly applied, we conclude that there is no underlying well-posed optimisation problem that the method solves. The underlying optimisation problem is of crucial importance in the theoretical study of variational regularisation methods, suggesting that the standard variational regularisation theory can not be applied to give theoretical guarantees for learned iterative reconstruction methods in general.

At the end of the previous chapter, we alluded to ways in which we could restrict the learned reconstruction method to move back towards the variational regularisation model. A standard

way to do this is to restrict to an algorithm taking a Plug-and-Play form (Chan, 2016; Chan et al., 2016; Sreehari et al., 2016; Venkatakrishnan et al., 2013), an example of which is shown in Algorithm 6 (here E_y is the data discrepancy as in the previous chapter). In this setting, the denoiser Φ can be a denoiser trained on natural images for a true Plug-and-Play algorithm, or the overall algorithm can be trained in an end-to-end fashion.

Algorithm 6 Plug-and-Play proximal gradient method

inputs: measurements y, initial estimate u^0, denoiser Φ
$u \leftarrow u^0$
for $i \leftarrow 1, \ldots, \mathtt{it}$ **do**
  $u \leftarrow \Phi(u - \tau^i \nabla E_y(u))$
end for
return u

Even with such algorithms, we may run into divergent behaviour if we do not restrict Φ appropriately (Sommerhoff et al., 2019), but there is recent work showing that the iterative method will converge as long as certain Lipschitz conditions are imposed on the denoiser Φ (Hertrich et al., 2020; Ryu et al., 2019).

The desire to impose Lipschitz conditions on neural networks has come to the forefront in a number of other tasks in recent years, especially because there have been serious concerns about the stability of neural networks ever since it was shown that high performance image classifiers may suffer from adversarial examples (Goodfellow et al., 2015). These issues need to be satisfactorily resolved before deep learning methods can be considered suitable for application in safety-critical systems. Another important application of Lipschitz neural networks can be found in generative modelling, in particular in models such as Wasserstein generative adversarial networks (GANs) (Arjovsky et al., 2017). In these models, the aim is to minimise the Wasserstein distance between the output of a generator neural network and some target distribution:

$$\min_{\Psi} W_1(\Psi \# \mu_{\text{latent}}, \mu_{\text{true}}), \tag{3.1}$$

where W_1 is the Wasserstein metric, μ_{latent} is a (simple) distribution of latent variables $Z \in \mathcal{Z}$, $\Psi \# \mu_{\text{latent}}$ is its pushforward by the generator neural network $\Psi : \mathcal{X} \to \mathcal{Z}$ and μ_{true} is the target distribution of $X \in \mathcal{X}$. Appealing to the Kantorovich-Rubinstein duality, we know that

$$W_1(\mu, \nu) = \sup_{f:\mathcal{X}\to\mathbf{R},\, 1\text{–Lipschitz}} \mathbb{E}_{X\sim\mu}[f(X)] - \mathbb{E}_{Y\sim\nu}[f(Y)],$$

where f is usually called the critic. With this result, Problem (3.1) becomes the following saddlepoint problem:

$$\min_{\Psi} \sup_{f:X\to\mathbf{R}\,1-\text{Lipschitz}} \mathbb{E}_{Z\sim\mu_{\text{latent}}}[f(\Psi(Z))] - \mathbb{E}_{X\sim\mu_{\text{true}}}[f(X)].$$

To solve this problem, we are required to flexibly parametrise 1-Lipschitz critic functions $f : \mathcal{X} \to \mathbf{R}$.

Lipschitz continuity is a standard way to quantify the stability of a function. Let us recall its definition and some associated properties: a function $f : \mathcal{X} \to \mathcal{Y}$ between metric spaces $\mathcal{X}$ and $\mathcal{Y}$ is said to be L-Lipschitz for some $L \geqslant 0$ if $d_{\mathcal{Y}}(f(x_1), f(x_2)) \leqslant L d_{\mathcal{X}}(x_1, x_2)$ for all $x_1, x_2 \in \mathcal{X}$. This notion of stability plays well with the compositional nature of neural networks: if $f_1 : \mathcal{X} \to \mathcal{Y}$ and $f_2 : \mathcal{Y} \to \mathcal{Z}$ are L_1-Lipschitz and L_2-Lipschitz respectively, their composition $f_2 \circ f_1$ is $(L_1 \cdot L_2)$-Lipschitz. If $\mathcal{X}$ and $\mathcal{Y}$ are in fact normed spaces, we can furthermore see (by definition) that any bounded linear operator $A : \mathcal{X} \to \mathcal{Y}$ is $\|A\|$-Lipschitz, where the norm is the operator norm. In particular, an ordinary feedforward neural network $\Psi(x) = \sigma(b^K + A^K\sigma(b^{K-1} + A^{K-1}\sigma(\ldots + A^2\sigma(b^1 + A^1x))))$ with a 1-Lipschitz activation function σ and learnable linear operators $A^1, \ldots, A^K$ and biases $b^1, \ldots, b^K$ is L-Lipschitz, where $L = \prod_{i=1}^{K} \|A^i\|$. This naturally gives rise to the idea of spectral normalisation: if an ordinary feedforward neural network with a given Lipschitz constant L is required for a specific application, this can be achieved by appropriately normalising the linear operators, as applied to GANs in Miyato et al. (2018). It is worth remarking here that we are denoting by Lipschitz constant of f any L that satisfies the defining inequality for Lipschitz continuity of f; often the term Lipschitz constant is used instead to refer only to the infimum of such L, which defines a seminorm on vector spaces of Lipschitz functions. We will refer to this infimum as the optimal Lipschitz constant of f, and note that the statements about the composition of Lipschitz functions, when framed in terms of optimal Lipschitz constants, only give upper bounds in general.

In this chapter we are focused on the case where $\mathcal{X}$ and $\mathcal{Y}$ are equal to each other, as is the case in many image-to-image tasks. Residual networks (ResNets) (He et al., 2016) have proven to be an extremely successful neural network meta-architecture in this setting: a ResNet parametrises a neural network by $\Psi = (\mathrm{id} + \Psi^K) \circ \ldots \circ (\mathrm{id} + \Psi^1)$, where each Ψ^i is a small neural network. Without further constraints, the Lipschitz continuity of such a network may be badly behaved as the depth increases: even if we control each Ψ^i to be ε-Lipschitz for some small $\varepsilon > 0$, in the worst case we can not guarantee anything better than that $\mathrm{id} + \Psi^i$ is $(1 + \varepsilon)$-Lipschitz, and that the composition Ψ is L-Lipschitz with $L = (1 + \varepsilon)^K$, which grows

exponentially as $K \to \infty$. Nevertheless, we show that it is possible to design ResNets that are provably nonexpansive (1-Lipschitz) by discretising nonexpansive continuous flows in a sufficiently careful manner (n.b. it is not guaranteed in general that a discretisation of a continuous flow preserves its structural properties, such as nonexpansiveness).

3.1.1 Related topics

Lipschitz neural networks

As mentioned above, within the deep learning community there have been a number of drivers for research into neural networks with controlled Lipschitz constants, such as the desire to increase robustness to adversarial examples, and the necessity to model the critic in a Wasserstein GAN as a 1-Lipschitz function. Spectral normalisation (Miyato et al., 2018) has become a standard approach to constraining the Lipschitz constant of an ordinary feedforward neural network. This approach ensures that the optimal (smallest) Lipschitz constant of a neural network is upper bounded. It is known to be computationally hard to estimate the true optimal Lipschitz constant (Virmaux and Scaman, 2018) of a neural network, which has prompted further research into refining Lipschitz neural network architectures.

Methods based on continuous dynamical systems

Applied mathematicians and physicists have long studied continuous dynamical systems in the form of ODEs and PDEs, giving rise to an extensive body of research on the structural properties of such systems. More recently, insights from these topics have been used to design neural network architectures which share similar structural properties (Chang et al., 2018; Ruthotto and Haber, 2020). The adjoint method for computing gradients has gained widespread use in the deep learning community, after it was shown in the Neural ODEs paper (Chen et al., 2018) that it is possible to parametrise the vector field defining an ODE by a neural network and differentiate through the flow to learn the vector field. This work has spawned a plethora of works that use learnable continuous dynamical systems.

Convex analysis and monotone operator theory

There is a recent line of work investigating the connections between existing deep learning practice and the topics of convex analysis and monotone operator theory. In particular, many of the standard activation functions that are used in neural networks are averaged (in the sense that we define in Section 3.2.1), and further analysis enables one to use this insight to

design neural networks that are averaged (Combettes and Pesquet, 2020; Hasannasab et al., 2020; Hertrich et al., 2020; Pesquet et al., 2020).

3.1.2 Our contributions

We describe and analyse a family of ResNet-styled neural network architectures that are guaranteed to be nonexpansive. The effect of these neural networks on input vectors can be thought of as sequentially composing parts of (discretisations of) gradient flows along learnable convex potentials. We show that it is only necessary to control the operator norms of the learnable linear operators contained in these networks to ensure their nonexpansiveness. This task is easily achieved in practice using power iteration.

The most basic such network takes the simple form described in Algorithm 7. For this network, we use convex analysis techniques to show that more fine-grained control of the learnable linear operators ensures that each layer of the network is averaged, and as a result that the overall network is averaged.

We demonstrate the use of the proposed architectures by studying their natural application to an image denoising task, focusing on the influence of various tunable aspects in the architectures for this problem, and comparing our approach to a standard approach to the denoising task.

3.2 Nonexpansive ODEs and the circle contractivity condition

Suppose that $f : \mathbf{R} \times \mathbf{R}^n \to \mathbf{R}^n$ is a time-dependent vector field and consider the ordinary differential equation (ODE) given by the flow along this vector field:

$$\dot{z}(t) = f(t, z(t)). \tag{3.2}$$

Assuming existence and uniqueness of the solutions to the ODE, we can define the flow map $\Psi : [0, \infty) \times \mathbf{R}^n \to \mathbf{R}^n$ by $\Psi(t, x) = z(t)$, where z solves Equation (3.2) with the initial condition $z(0) = x$. Since the vector fields that we will consider are (globally) Lipschitz continuous, global existence and uniqueness is not an issue by the Picard-Lindelöf theorem (Teschl, 2012). It is natural to ask when this flow map is nonexpansive, in the sense that $\|\Psi(t, x) - \Psi(t, y)\| \leqslant \|x - y\|$ for all t, x, y. Letting $t \to 0$, we see that it is necessary that

$$\langle f(t, x) - f(t, y), x - y \rangle \leqslant 0,$$

and conversely, if this condition holds the flow map is nonexpansive since

$$\frac{\mathrm{d}}{\mathrm{d}t}\|\Psi(t, x) - \Psi(t, y)\|^2 = 2\langle f(t, x(t)) - f(t, y(t)), x(t) - y(t) \rangle \leqslant 0. \tag{3.3}$$

In practice, most ODEs of interest are not explicitly solvable and it is necessary to resort to numerical methods to approximate the flow map. A very well-studied class of such numerical integrators is the class of Runge-Kutta methods, which can be defined as follows:

Definition 3.2.1 (Runge-Kutta method)**.** If $m \in \mathbf{N}$ is a positive integer, an m-stage Runge-Kutta (RK) method is characterised by a matrix $a \in \mathbf{R}^{m \times m}$ and two vectors $b, c \in \mathbf{R}^m$, satisfying $\sum_{j=1}^m b_j = 1$ and $c_i = \sum_{j=1}^m a_{i,j}$. For a step size $h > 0$, the RK method approximates the step from $y = \Psi(t, x)$ to $\Psi(t + h, x)$ as follows:

$$\Phi_h(t, y, f) = y + h \sum_{i=1}^m b_i f(t + c_i h, Y_i),$$

where Y solves the nonlinear equations

$$Y_i = y + h \sum_{j=1}^m a_{ij} f(t + c_j h, Y_j).$$

If a is strictly lower triangular, these equations are solvable in a single pass and the method is called explicit. Otherwise, the method is called an implicit method.

Since we have the goal of designing neural networks that encode nonexpansive operators, it is of particular interest to know whether a given numerical integrator preserves the nonexpansiveness of a continuous flow for which Inequality (3.3) holds. This property of a numerical integrator is called BN-stability and has been studied in detail for RK methods in Burrage and Butcher (1979); for these methods, BN-stability is equivalent to algebraic stability, which is defined by a simple algebraic condition on the coefficients on the method. A comprehensive overview of stability properties for RK methods is given in Hairer and Wanner (1996). It is well known (see for instance Nevanlinna and Sipilä (1974)) however that no explicit RK method can satisfy such an unconditional stability condition. Nevertheless, it was shown in Dahlquist and Jeltsch (1979) that a conditional stability result can be established for certain explicit RK methods as long as Inequality (3.3) is replaced by an alternative that has the effect of controlling the stiffness of the ODE. To state this result, we require the definition of the circle contractivity property of an RK method.

Definition 3.2.2 (Circle contractivity)**.** Suppose that $a \in \mathbf{R}^{m\times m}$ and $b, c \in \mathbf{R}^m$ are the matrix and vectors characterising an RK method as in Definition 3.2.1. We say that this RK method satisfies the r-circle contractivity condition for a given $r \in \mathbf{R} \cup \{\infty\}$ if $|K(\zeta)| \leqslant 1$ for all $\zeta \in D(r)^m$. Here, the function $K : \mathbf{C}^m \to \mathbf{C}$ can be thought of as the action of the method on a nonautonomous linear ODE:

$$K(\zeta) = 1 + b^T \operatorname{diag}(\zeta)(\mathrm{id} - a \operatorname{diag}(\zeta))^{-1}\mathbf{1},$$

and $D(r)$ is a generalised disk:

$$D(r) = \begin{cases} \{z \in \mathbf{C} \,|\, |z + r| \leqslant r\} & \text{when} \quad r \geqslant 0, \\ \{z \in \mathbf{C} \,|\, \operatorname{Re}(z) \leqslant 0\} & \text{when} \quad r = \infty, \\ \{z \in \mathbf{C} \,|\, |z + r| \geqslant -r\} & \text{when} \quad r < 0. \end{cases}$$

Example 3.2.1. The most basic nontrivial example of an RK method is the forward (explicit) Euler method, given by $\Phi_h(t, y, f) = y + hf(t, y)$. In the notation of Definition 3.2.1, we have $m = 1$, $a = 0$ and $b = 1$, so $K(z) = 1 + z$. We conclude that the forward Euler method is 1-circle contractive.

Remark 3.2.1. It is straightforward to compute the optimal (in the sense that it gives the largest generalised disk) r for which a given RK method is r-circle contractive if we know

a and b: if we define the symmetric matrix $Q = \operatorname{diag}(b)a + a^T \operatorname{diag}(b) - bb^T$, Theorem 3.1 from Dahlquist and Jeltsch (1979) tells us that $r = -1/\rho$, where ρ is the largest number such that $w^T Q w \geqslant \rho w^T \operatorname{diag}(b) w$ for all $w \in \mathbf{R}^m$. Hence, if we can solve the generalised eigenvalue problem $Qv = \lambda \operatorname{diag}(b)v$, we know that the minimal eigenvalue gives the desired ρ.

With this definition, it is now possible to state the conditional stability result that extends to certain explicit methods:

Theorem 3.2.1 (Theorem 4.1 from Dahlquist and Jeltsch (1979))**.** *Suppose that Φ_h is an RK method satisfying the r-circle contractivity condition, and that f satisfies the monotonicity condition*

$$\langle f(t,y) - f(t,z), y - z\rangle \leqslant -\nu \|f(t,y) - f(t,z)\|^2. \tag{3.4}$$

Then, if $r \neq \infty$ and $h/r \leqslant 2\nu$, or if $r = \infty$ and $\nu \geqslant 0$,

$$\|\Phi_h(t,y,f) - \Phi_h(t,z,f)\| \leqslant \|y - z\|.$$

The idea of using this result to design nonexpansive neural networks was recently discussed in Celledoni et al. (2021a), though in this work no indication was given of how the vector fields should be parametrised. The monotonicity condition given by Inequality (3.4) is reminiscent of the property of co-coercivity, known mainly from the theory of convex optimisation for its use in the Baillon-Haddad theorem:

Theorem 3.2.2 (Corollary 18.16 from Bauschke and Combettes (2011))**.** *Suppose that $\phi : \mathcal{X} \to \mathbf{R}$ is a Fréchet-differentiable convex function on a Hilbert space $\mathcal{X}$. Then ϕ is L-smooth for some $L \geqslant 0$ (equivalently, $\nabla\phi$ is L-Lipschitz), meaning that*

$$\phi(y) \leqslant \phi(x) + \langle \nabla\phi(x), y - x\rangle + \frac{L}{2}\|y - x\|^2,$$

if and only if $\nabla\phi$ is $1/L$-co-coercive, meaning that

$$\langle \nabla\phi(y) - \nabla\phi(x), y - x\rangle \geqslant \frac{1}{L}\|\nabla\phi(y) - \nabla\phi(x)\|^2.$$

Indeed, if $f(t,x) = -\nabla\phi(x)$ for a $1/\nu$-smooth convex potential $\phi : \mathbf{R}^n \to \mathbf{R}$ (so that we have a gradient flow of a smooth convex potential), then Inequality (3.4) is satisfied. This connection has recently been used to demonstrate in Sanz Serna and Zygalakis (2020) that there is an explicit Runge-Kutta method for which the circle contractivity disk degenerates to a point, by constructing a smooth convex potential for which the nonexpansiveness of the flow map is not preserved.

For the purpose of using this observation and Theorem 3.2.1 to design nonexpansive neural networks, note the following result:

Lemma 3.2.1. *Suppose that $\sigma : \mathbf{R} \to \mathbf{R}$ is an increasing L-Lipschitz activation function, $A : \mathbf{R}^{n\times k}$ is a matrix and $b \in \mathbf{R}^n$ is a bias vector. The vector field $f_{A,b}(t,x) = -A^T\sigma(Ax+b)$ (where σ is applied separately to each component) satisfies Inequality (3.4) with $\nu = 1/(\|A\|^2 L)$.*

Proof. Since σ is increasing and L-Lipschitz, the function $\psi : \mathbf{R} \to \mathbf{R}$ given by $\psi(t) = \int_0^t \sigma(s)\,\mathrm{d}s$ is convex and L-smooth. Hence, $\phi : \mathbf{R}^n \to \mathbf{R}$ given by

$$\phi(x) = \sum_{i=1}^{n} \psi(x_i)$$

is convex and L-smooth. The functional $x \mapsto \phi(Ax+b)$ is convex and by the chain rule it has gradient equal to $-f_{A,b}$ and it is $\|A\|^2 L$-smooth. By the comments preceding this lemma, the vector field $f_{A,b}$ satisfies Inequality (3.4) with $\nu = \|A\|^2 L$. □

By the previous observations, we can propose a natural nonexpansive neural network architecture as follows: given an $r > 0$ such that we have an r-circle contractive RK method Φ_h and an L-Lipschitz increasing activation function σ, consider linear operators $A^1, \ldots, A^{\mathtt{it}}$, biases $b^1, \ldots, b^{\mathtt{it}}$ and stepsizes $h^1, \ldots, h^{\mathtt{it}}$ and define the operator Ξ by

$$\Xi = \Xi^{\mathtt{it}} \circ \ldots \circ \Xi^1,$$

where $\Xi^i(x) = \Phi_{h^i}(0, x, f_{A^i,b^i})$ is one numerical integration step along the vector field f_{A^i,b^i} as defined in Lemma 3.2.1. Lemma 3.2.1 and Theorem 3.2.1 ensure that Ξ is nonexpansive as long as $h^i\|A^i\|^2 \leqslant 2r$. There are various ways in which this bound can be maintained during training, and the power method can be used to compute the required operator norm: as an example, it is possible to alternate gradient update steps of an optimiser with steps that scale the operators down to satisfy the bounds that are violated after the gradient update.

For any explicit RK method, the corresponding neural network Ξ is a residual network. For the forward Euler method, the network takes the following particularly simple form:

As mentioned before, we are focused in this chapter on explicit RK methods since they do not require the solution of a (potentially difficult) nonlinear equation at each step. It may be interesting to note, however, what can happen when an implicit numerical method is used, such as the backward Euler method. In that case, each update step in Algorithm 7 needs to be

Algorithm 7 Forward Euler method for nonexpansive ODE networks

input: vector x
parameters: a step size $0 < h^i \leqslant 2$, linear operators $A^1, \dots, A^{\mathtt{it}}$ satisfying $\|A^i\| \leqslant 1/L$ for $i = 1, \dots, \mathtt{it}$, and biases $b^1, \dots, b^{\mathtt{it}}$
$z^0 \leftarrow x$
for $i \leftarrow 1, \dots, \mathtt{it}$ **do**
    $z^i \leftarrow z^{i-1} - h^i (A^i)^T \sigma(A^i z^{i-1} + b^i)$
end for
return $\Xi(x) = z^{\mathtt{it}}$

replaced by solving the equation

$$z^i = z^{i-1} - h^i f_{A,b}(z^i) = z^{i-1} - h^i (A^i)^T \sigma(A^i z^i + b^i).$$

Recalling from the proof of Lemma 3.2.1 that $-f_{A,b}$ is the gradient of a convex functional $\phi(A \cdot + b)$, this shows that the update step is given by

$$z^i = (\mathrm{id} + h^i \nabla \phi(A \cdot + b))^{-1}(z^{i-1}) =: \mathrm{prox}_{h^i \phi(A \cdot + b)}(z^{i-1}),$$

which is the defining equation of the proximal operator (Moreau, 1963), a mathematical object that has been studied in great detail in the field of convex analysis. Whether considering it from the ODE viewpoint (the backward Euler method is BN-stable (Burrage and Butcher, 1979)) or from the convex analysis and monotone operator theory viewpoint (proximal operators are nonexpansive as the resolvents of monotone operators (Bauschke and Combettes, 2011, Chapter 23)), proximal operators $\mathrm{prox}_{h\phi(A \cdot + b)}$ are well-defined and nonexpansive regardless of the step size $h > 0$ and the smoothness of $\phi(A \cdot + b)$. This unconditional stability comes at a cost, though: for general A, computing the proximal operator of $\mathrm{prox}_{h\phi(A \cdot + b)}$ is not easy (and becomes more difficult as $\|A\|$ increases). This issue can be overcome by restricting A to certain special sets of operators (for instance satisfying certain orthogonality properties), in which case the proximal operator may be explicitly computable. This approach is similar to the one taken in Hasannasab et al. (2020) and Hertrich et al. (2020), though note that it may be difficult to enforce these constraints on convolution-type linear operators. On the other hand, the operator norm constraints that we are required to enforce with explicit numerical integration methods can be easily controlled using power iteration (Golub and Vorst, 2000); all we need is the ability to apply the operator and its adjoint to test vectors.

3.2.1 A more detailed look at the architecture for the forward Euler method

When the numerical integrator used is the forward Euler method, as described in Algorithm 7, straightforward computations can be used to establish the same results guaranteed by the machinery of Theorem 3.2.1, and some more nuanced results. Indeed, it is possible to choose the stepsizes in such a way that the resulting neural network is not just nonexpansive, but in fact is also averaged:

Definition 3.2.3 (Definition 4.23 from Bauschke and Combettes (2011))**.** Suppose that $A : \mathcal{X} \to \mathcal{X}$ is an operator mapping a Hilbert space $\mathcal{X}$ into itself and that $\alpha \in (0,1)$. We call A an α-averaged operator if there is a nonexpansive $T : \mathcal{X} \to \mathcal{X}$ such that $A = (1-\alpha)\,\mathrm{id} + \alpha T$. We may also leave α unspecified, in which case we just call A an averaged operator if there is an $\alpha \in (0,1)$ such that A is α-averaged.

Note that the triangle inequality shows that an averaged operator is nonexpansive. In addition, averaged operators allow for convergent fixed point iterations, whereas ordinarily nonexpansive operators enjoy no such guarantees. This is of crucial importance in certain applications, such as Plug-and-Play algorithms, where modelling denoisers using nonexpansive operators is not enough to prevent divergence, but using averaged operators can ensure convergence (Hertrich et al., 2020). For our analysis here, let us note the following fact:

Lemma 3.2.2. *Suppose that $\Xi : \mathbf{R}^n \to \mathbf{R}^n$ is C^1 with symmetric Jacobian everywhere and that $\alpha \in (0,1)$. Then Ξ is α-averaged if and only if*

$$\mathrm{spectrum}(D\Xi(x)) \subset [1-2\alpha, 1]$$

for all $x \in \mathbf{R}^n$. Note that the condition that the Jacobian is everywhere symmetric is equivalent to asking that $\Xi = \nabla f$ for some underlying functional $f : \mathbf{R}^n \to \mathbf{R}^n$.

Recall that a single layer of the proposed architecture is given by $\Xi(x) = x - hA^*\sigma(Ax+b)$, with the same setting in mind as described in Lemma 3.2.1. There we saw that Ξ is the gradient of the functional $x \mapsto \|x\|^2/2 - h\phi(Ax+b)$, where ϕ is convex and L-smooth, so that $\mathrm{spectrum}(D^2\phi(x)) \subset [0, L]$ for each $x \in \mathbf{R}^n$. Hence, since we have $D\Xi(x) = \mathrm{id} - hA^*D^2\phi(Ax+b)A$, we find that

$$\mathrm{spectrum}(D\Xi(x)) \subset [1 - h\|A\|^2 L, 1].$$

Combining this with Lemma 3.2.2 immediately gives the following result if the activation function σ is C^1. This is not required, however, for the result to be valid; any L-Lipschitz σ,

such as $\sigma = \texttt{ReLU}$, will work equally well. The argument for general L-Lipschitz σ is given below:

Theorem 3.2.3. *Let σ, A, b be as in Lemma 3.2.1 and let $\alpha \in (0,1)$. A single layer of the proposed architecture, $\Xi(x) = x - hA^*\sigma(Ax+b)$, is α-averaged if $h\|A\|^2 \leqslant 2\alpha/L$.*

Proof. The argument given above provides some intuition regarding averaged operators, but requires the activation function σ to be C^1. Here, we will show that this is not necessary. Indeed, note that an operator Ξ is α-averaged if and only if $(\Xi - \mathrm{id})/\alpha + \mathrm{id}$ is nonexpansive. Furthermore, we note that $\Xi - \mathrm{id} = hf_{A,b}$, where $-f_{A,b}$ is the gradient of an $\|A\|^2 L$-smooth convex functional, as defined in the proof of Lemma 3.2.1. By Theorem 3.2.2 we have that

$$\frac{1}{\|A\|^2 L}\|f_{A,b}(x) - f_{A,b}(y)\|^2 \leqslant \langle -f_{A,b}(x) + f_{A,b}(y), x - y\rangle \leqslant \|A\|^2 L\|x-y\|^2,$$

so, if we write $\Lambda = (\Xi - \mathrm{id})(x) - (\Xi - \mathrm{id})(y)$ to reduce clutter, we have

$$-h\|A\|^2 L\|x-y\|^2 \leqslant \langle \Lambda, x - y\rangle \leqslant -\frac{1}{h\|A\|^2 L}\|\Lambda\|^2.$$

In particular, $\langle \Lambda, x-y\rangle + \|\Lambda\|^2/(h\|A\|^2 L) \leqslant 0$. Upon expanding the squared norm, we find that

$$\begin{aligned}\|((\Xi - \mathrm{id})/\alpha + \mathrm{id})(x) - ((\Xi - \mathrm{id})/\alpha + \mathrm{id})(y)\|^2 &\leqslant \frac{\|\Lambda\|^2}{\alpha^2} + 2\frac{\langle \Lambda, x-y\rangle}{\alpha} + \|x-y\|^2 \\ &\leqslant \|x-y\|^2 + \frac{2}{\alpha}\left(\langle \Lambda, x-y\rangle + \frac{\|\Lambda\|^2}{2\alpha}\right).\end{aligned}$$

By the above comments, we see that $(\Xi - \mathrm{id})/\alpha + \mathrm{id}$ is nonexpansive when $2\alpha \geqslant h\|A\|^2 L$, which can be rewritten into $h\|A\|^2 \leqslant 2\alpha/L$. □

Furthermore, the following result guarantees that the overall network will be averaged as long as each layer is averaged, with a corresponding α that can be controlled:

Theorem 3.2.4 (Proposition 4.32 from Bauschke and Combettes (2011))**.** *Suppose that $\Xi^1, \ldots, \Xi^m$ are operators $\Xi^i : \mathcal{X} \to \mathcal{X}$ on a Hilbert space $\mathcal{X}$ and that each Ξ^i is α_i-averaged for some $\alpha_i \in (0,1)$. Then $\Xi^m \circ \ldots \circ \Xi^1$ is α-averaged, where*

$$\alpha = \frac{m}{m - 1 + \min\limits_{i=1,\ldots,m}(1/\alpha_i)}.$$

In particular, if we are targeting a certain $\alpha \in (0,1)$ for which our neural network ($\mathtt{it}$ layers deep) should be α-averaged, we should ask that each layer is α_i-averaged with α_i at most

$$\alpha_i \leqslant \frac{\alpha}{\mathtt{it}(1-\alpha)+\alpha}.$$

By Theorem 3.2.3, we see that this implies that we must use a step size $h = O(1/\mathtt{it})$ that decreases to 0 as the depth $\mathtt{it}$ of the network increases. Alternatively, it is possible to get an averaged operator by appealing to Definition 3.2.3: $(1-\alpha)\,\mathrm{id} + \alpha\Xi$ will be α-averaged as long as Ξ is nonexpansive, which we have seen can be guaranteed with a step size independent of the depth of the network.

3.3 Experiments

In all the experiments that we describe here, except the ones using the higher order numerical integrator in Section 3.3.3, we use the architecture described in Algorithm 7 with the activation function σ chosen to be the rectified linear unit $\texttt{ReLU}(x) = (x)_+$, which is 1-Lipschitz. We keep the stepsizes fixed and equal for each layer in the network, at a value motivated by the results in Section 3.2 depending on whether we are modelling a nonexpansive operator or an operator that is also averaged.

We train each networks in a supervised manner, by attempting to solve an empirical risk minimisation problem. As mentioned before, we use the power iteration method (Miyato et al., 2018) to compute spectral norms of each of the learnable linear operators: if $A : \mathbf{R}^n \to \mathbf{R}^m$ is a linear operator and initial estimate of the first left singular vector $v^0 \in \mathbf{R}^m$, we iterate

$$u^k \leftarrow \frac{A^T v^{k-1}}{\|A^T v^{k-1}\|}, \qquad v^k \leftarrow \frac{A u^k}{\|A u^k\|}.$$

Assuming that v^0 is not orthogonal to the first left singular vector (this is guaranteed to hold with probability 1 if v^0 is randomly selected from a probability distribution that has a density w.r.t. Lebesgue measure), u^k and v^k converge to the first singular vectors of A as $k \to \infty$ and $(u^k)^T A v^k \to \|A\|$.

It is possible to differentiate through the spectral normalisation step $A \mapsto A/\|A\|$ (Miyato et al., 2018), but we find that a simpler approach also works to enforce the operator norm constraints: to train the networks we alternate gradient update steps (using the Adam optimisation method (Kingma and Ba, 2017) with a fixed learning rate of 10^{-4}) with normalisation steps, in which we use the power method to check whether the operator norms of the linear operators exceed the bounds given in Algorithm 7. Where the bounds are violated, we normalise by dividing by the appropriate multiple of the current norm. As described in Miyato et al. (2018), it is possible to use the estimate of the left first singular vector output by the preceding application of the power method to warm-start the current application of the power method, and it is generally sufficient to perform just a single iteration of the power method when warm-started in this way.

To initialise the networks, we first use the He initialisation method (He et al., 2015) to initialise the convolutional filters, and apply 1000 iterations of the power method to compute their norms. The filters are then normalised to satisfy the required bounds and the singular vectors output by the power method are saved for future iterations. For each bias vector,

we randomly initialise with a Gaussian white noise vector normalised so that the expected squared norm equals 1.

All experiments have been implemented using PyTorch (Paszke et al., 2019) and using NumPy (Harris et al., 2020). Each experiment was run on a single computational node with an Intel Xeon Gold 6140 CPU and a NVIDIA Tesla P100 GPU.

3.3.1 A toy example

To warm up and gain an understanding of some of the benefits that may be had from using a nonexpansive ResNet architecture over an unconstrained ResNet, consider the following toy problem. We aim to approximate the absolute value function $|\cdot| : \mathbf{R} \to \mathbf{R}$ given a tiny training set consisting of just 6 random points at which we know the exact function value. We train the nonexpansive architecture Ξ of Algorithm 7, with $h = 2, \mathtt{it} = 10$ and $A^i \in \mathbf{R}^{10\times 1}, b^i \in \mathbf{R}^{10}$. To compare, we consider the comparable ordinary ResNet architecture Γ, where the update steps are replaced by $z^i \leftarrow z^{i-1} + B^i\sigma(A^i z^{i-1} + b^i)$ with $b^i \in \mathbf{R}^{10}$ and $A^i \in \mathbf{R}^{10\times 1}, B^i \in \mathbf{R}^{1\times 10}$ unconstrained. Note that Γ can replicate the action of Ξ by setting $B^i = -h(A^i)^T$ and appropriately constraining the weights, but Γ is strictly more flexible than Ξ. Both architectures are trained to minimise the squared error on the training set and achieve (up to machine precision) perfect reconstruction on the training set, but as seen in Figure 3.1 the unconstrained ResNet fails to be nonexpansive away from the training points, in contrast to the behaviour of the nonexpansive ResNet.

3.3.2 Nonexpansive neural networks for denoising

We use the BSDS500 dataset (Arbelaez et al., 2011), which is freely available under the GNU AGPL, as training data and test data for our denoising experiments. This dataset consists of 500 RGB images, split into $N_{\text{train}} = 200$ training images, $N_{\text{val}} = 100$ validation images and $N_{\text{test}} = 200$ test images. In our experiments, we adhere to the same splitting of the dataset. We scale the images so that each channel only contains values in $[0, 1]$ and simulate noisy images y corresponding to each ground truth image x^* by adding Gaussian white noise ε with a standard deviation of 0.5. The architectures that we consider for this task are of the form $\Gamma = A_{\text{project}} \circ \Xi \circ A_{\text{lift}}$, where A_{lift} is a convolution taking the 3 input channels to 64 channels, A_{project} is a convolution taking 64 channels to the 3 output channels, and Ξ is a network as in Algorithm 7 with each A^i a convolution taking 64 channels to 64 channels and each $b^i \in \mathbf{R}^{64}$. All convolution operators have kernel size 3×3. To ensure that Γ is nonexpansive, we are of

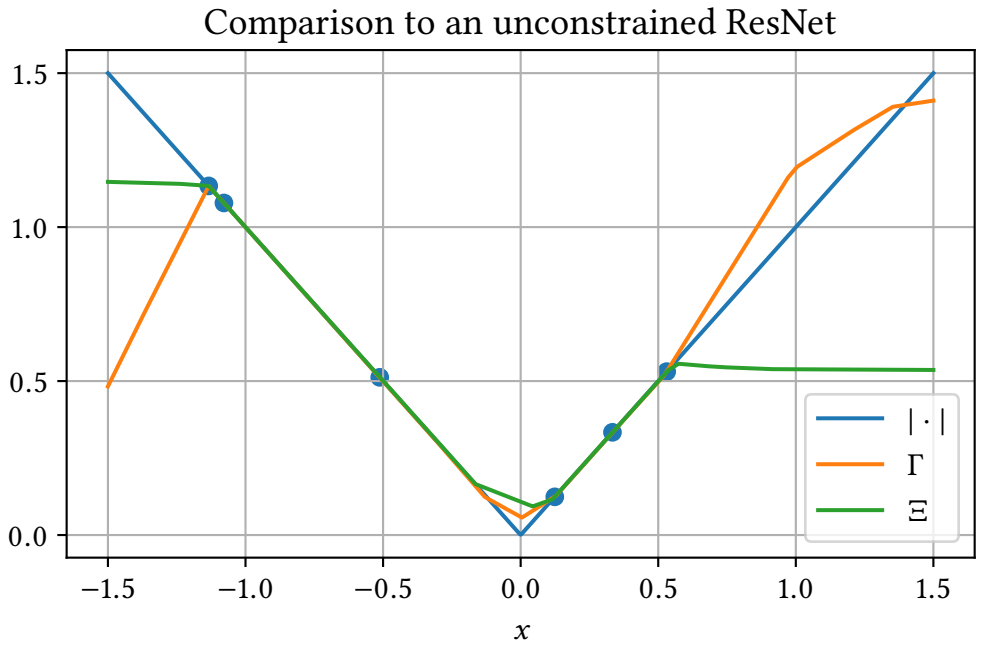

Figure 3.1: A comparison between the nonexpansive ResNet Ξ and a comparable unconstrained ResNet Γ on the problem of approximating the absolute value function given a small training set.

course required to enforce an operator norm bound on A_{project} and A_{lift} in addition to those that are required for Ξ.

As was alluded to in Hertrich et al. (2020), it is difficult to train Γ to map noisy images y to corresponding ground truth images u^*, but easier to use the residual learning approach (Zhang et al., 2017a). Furthermore, we can study the effect of multiplying Γ by a scaling parameter $\gamma \geqslant 1$, which increases the guaranteed Lipschitz constant to be γ, allowing the network to be less constrained. We will look at scaled denoisers trained in the ordinary (nonresidual) way:

$$\min_{\Gamma = A_{\text{project}} \circ \Xi \circ A_{\text{lift}}} \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \|\gamma \Gamma(y_i) - u_i^*\|^2, \tag{3.5}$$

in which case the denoiser is given by $\gamma\Gamma$ after training. We will also look at scaled denoisers trained in the residual way:

$$\min_{\Gamma = A_{\text{project}} \circ \Xi \circ A_{\text{lift}}} \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \|\gamma \Gamma(y_i) - (y_i - u_i^*)\|^2, \tag{3.6}$$

in which case the denoiser is given by $\mathrm{id} - \gamma\Gamma$ after training. We train each denoiser for 2000 epochs, using minibatches of size 5, unless otherwise specified.

The DnCNN, introduced in Zhang et al. (2017a), has become a standard benchmark for denoising tasks. A natural comparison to make is between our network Γ with $h = 2$ and $\texttt{it} = 10$, and the DnCNN $\Gamma_{\mathrm{DnCNN}} = A_{\mathrm{project}} \circ \Xi_{\mathrm{DnCNN}} \circ A_{\mathrm{lift}}$ where Ξ_{DnCNN} is a 20-layer convolutional neural network without skip connections, the details of which are described in Zhang et al. (2017a). Indeed, note that each layer of our architecture contains a convolution and its transpose, whereas the DnCNN uses one convolution per layer. This is trained to solve Problem (3.6) with $\gamma = 1$, in the same way as our architectures except that no operator norm constraints are enforced on the convolutions.

As another benchmark, we can consider total variation (TV) denoising, which gives the denoised image as

$$\hat{u} = \operatorname*{argmin}_u \frac{1}{2}\|u - y\|^2 + \alpha\|\nabla u\|_1,$$

where we have tuned α for optimal reconstruction performance on the training set.

Comparing all of these options, we observe the results shown in Figure 3.2: as the scaling parameter γ increases, both the nonresidual and residual methods approach the performance of DnCNN, which is to be expected since DnCNN is unconstrained. The significance of the scaling parameter $\gamma = 1.99$ is that γ needs to be kept below 2 if one wants to apply the "oracle trick" described in Hertrich et al. (2020) to obtain an averaged operator for use in a provably convergent Plug-and-Play algorithm. Figure 3.3 shows corresponding reconstructions on a test image. Evidently, the unscaled denoisers (i.e. for $\gamma = 1$) do not perform well enough to be considered acceptable, but the scaled residual denoiser with $\gamma = 1.99$ makes a reasonable tradeoff between stability and reconstruction quality.

3.3.3 Higher-order integrators

Although the concrete architecture obtained when using the forward Euler integrator (as described in Algorithm 7) is appealing in its simplicity, the framework laid out in Section 3.2 also allows us to use certain higher order integrators. For instance, consider Heun's method, which is given by

$$\Phi_h^{\mathrm{Heun}}(t, y, f) = y + \frac{h}{2}\Big(f(t, y) + f(t + h, y + hf(t, y))\Big). \tag{3.7}$$

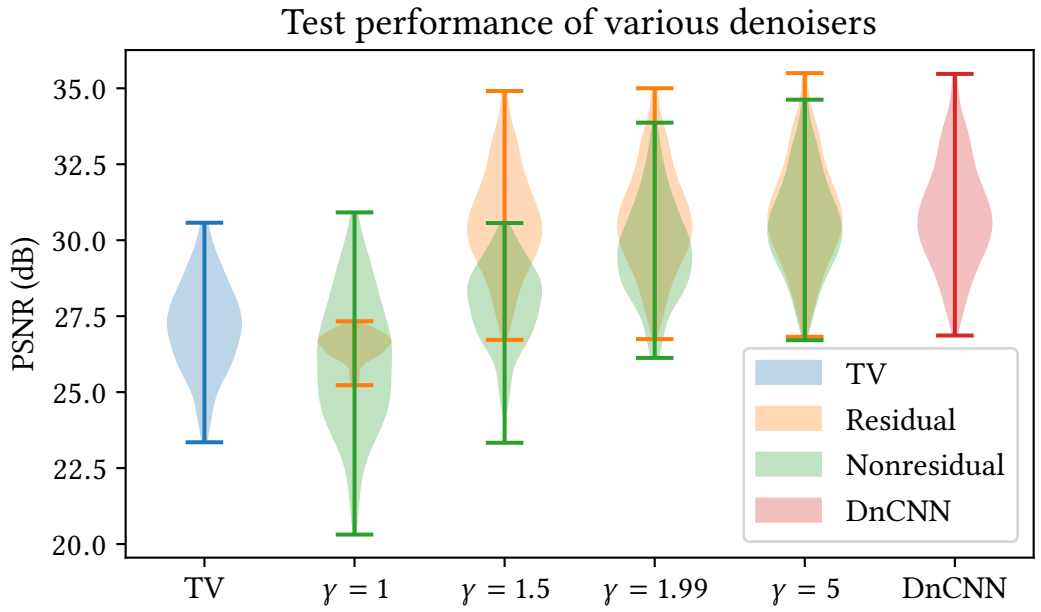

Figure 3.2: A comparison of the test performance of denoising by DnCNN, TV denoising and denoising using the scaled nonexpansive operators in a residual and a nonresidual way.

This is a 2-stage, second-order, RK method, with

$$a = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}, \qquad \operatorname{diag}(b)a + a^T \operatorname{diag}(b) - bb^T = \frac{1}{4}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix},$$

and using Remark 3.2.1, we conclude that Heun's method is 1-circle contractive, just as the forward Euler method. As a result, $x \mapsto \Phi_h^{\text{Heun}}(0, x, f_{A,b})$ is nonexpansive as long as $h\|A\|^2 \leqslant 2$ and Algorithm 7 can be adapted to use Heun's method, the only change being that the steps $z^i \leftarrow z^{i-1} - h(A^i)^T\sigma(A^i z^{i-1} + b^i)$ are replaced by steps of the form $z^i \leftarrow \Phi_h^{\text{Heun}}(0, z^{i-1}, f_{A^i,b^i})$. Fixing again $h = 2$ and $\texttt{it} = 10$, and training the architecture with Φ^{Heun} as the numerical integrator, we are required to reduce the minibatch size to fit in memory.

Similarly, we can consider integrators with yet higher orders, such as the fourth-order RK4 integrator Φ_h^{RK4}, given by Definition 3.2.1 with

$$a = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{6} \end{pmatrix}, \qquad \operatorname{diag}(b)a + a^T \operatorname{diag}(b) - bb^T = \frac{1}{9}\begin{pmatrix} -\frac{1}{4} & 1 & -\frac{1}{2} & -\frac{1}{4} \\ 1 & -1 & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & -1 & 1 \\ -\frac{1}{4} & -\frac{1}{2} & 1 & -\frac{1}{4} \end{pmatrix}.$$

Again using Remark 3.2.1, we conclude that the RK4 method is 1-circle contractive and we can replace the forward Euler method in Algorithm 7 by the RK4 method to obtain a nonexpansive neural network.

We train a scaled residual denoiser with $\gamma = 1.99$ using Heun's integrator and another using the RK4 method and compare to the result we obtained with the forward Euler method. Besides being computationally more intensive, we see in Table 3.1 that denoising performance actually suffers a bit from using the higher order methods as opposed to the forward Euler method, suggesting that there is no benefit to using a higher order numerical integrator on this task.

Table 3.1: A comparison of the means and standard deviations of the PSNRs computed on the test set, for the architecture using the forward Euler method, the architecture using Heun's method and the architecture using the RK4 method.

	forward Euler	Heun	RK4
PSNR (dB)	30.69 ± 1.69	29.48 ± 1.34	29.76 ± 1.43

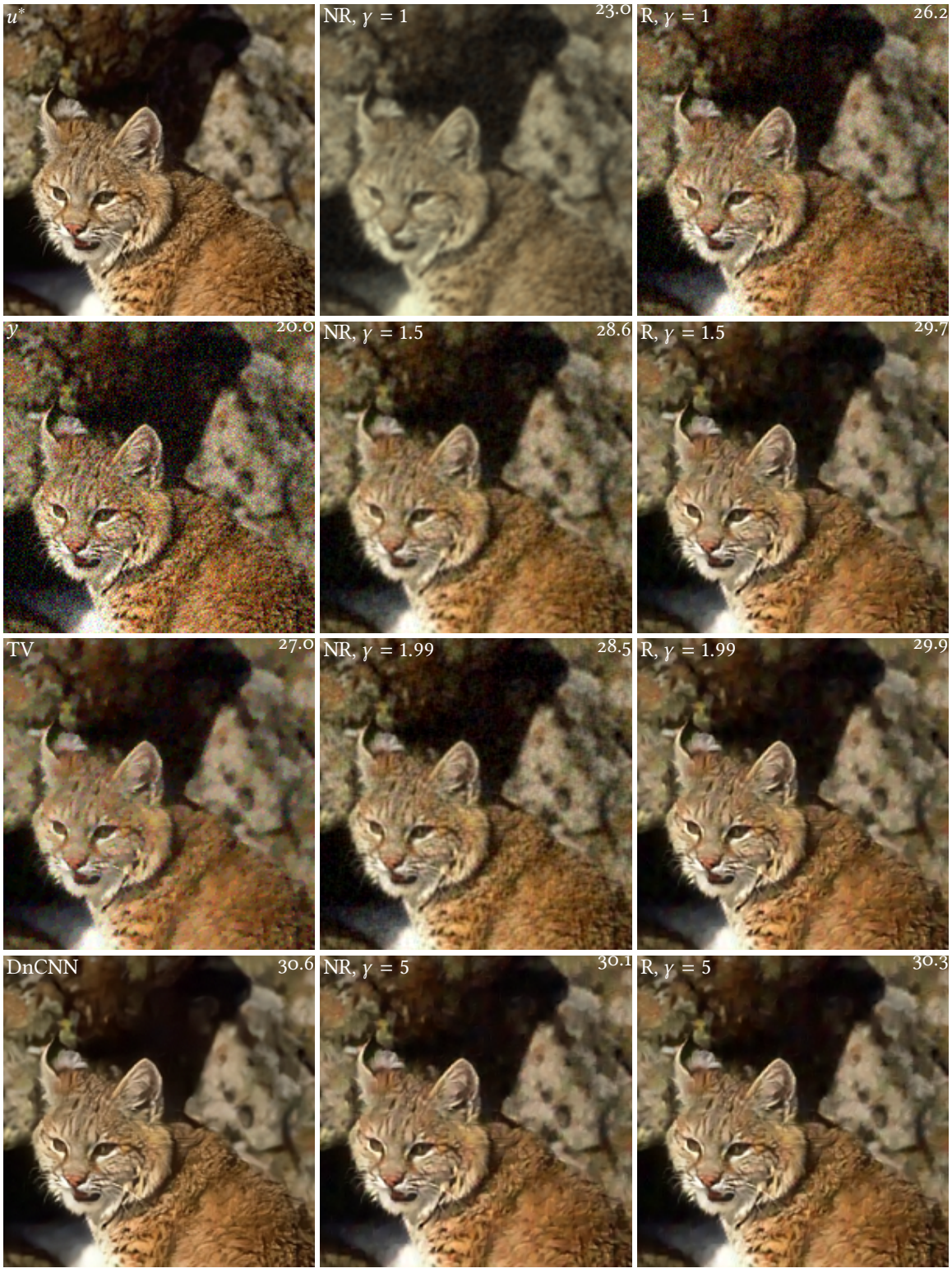

Figure 3.3: A comparison of zoomed in regions of test reconstructions for the methods compared in Figure 3.2. R indicates the scaled nonexpansive network used in a residual way, whereas NR indicates the scaled nonexpansive network used in a nonresidual way. The images on the right are zoomed in regions of one of the test images, with the numbers in the top right corner being the PSNR.

3.4 Conclusions and discussion

We have exhibited a family of ResNet architectures for which it is straightforward to enforce nonexpansiveness. The proposed architecture is given by compositions of numerical integration steps along gradient flows in convex potentials. For the main example using the forward Euler method, we have used tools from convex analysis to show that the architecture can be used to encode averaged operators. We have demonstrated the use of the proposed architectures on a denoising task. Notably, a combination of scaling and residual learning needs to be used to obtain denoisers that reach a reconstruction quality close to the state-of-the-art, as was observed for a similar model in Hertrich et al. (2020). Although the basic architecture uses the first order forward Euler method as the numerical integrator, it is possible to use higher order methods. Future work may study the application of these architectures in typical deep learning applications such as GANs, specifically in Wasserstein GANs which require the use of a 1-Lipschitz critic function. We have seen in practice that the proposed architecture is quite expressive, but an interesting direction for future work would be to study ways in which more general learnable nonexpansive flows can be used to motivate the design of provably stable neural network architectures, and provide approximation guarantees for them.

Conclusions and future work

In this dissertation, we have investigated various ways in which desirable structure can be incorporated into machine learning methods for inverse problems, and have observed benefits that can be had from doing so in a principled way:

- Using a bilevel learning approach, we can jointly learn sampling patterns and regularisation parameters to obtain better variational reconstructions than obtained using the standard random sampling patterns.
- The proximal operators in a learned iterative reconstruction method are naturally modelled as roto-translationally equivariant rather than just translationally equivariant (as is usually done). This results in lower sample complexity, allowing for better performance with smaller training sets, and more robustness of the reconstructions to rotations of the underlying image.
- Drawing on connections to convex analysis we can design ResNets that are provably nonexpansive, and even averaged, operators. These neural networks are natural candidates for applications that require stability.

As we speak, learned iterative reconstruction methods, such as the ones we discussed in Chapter 2, are considered to represent the state of the art in reconstruction methods for inverse problems. Given sufficient amounts of data they can perform significantly better at image reconstruction than variational regularisation methods with hand-crafted priors, and they do so while expending much less computational effort. However, as we saw in Figure 2.12 and discussed in Section 3.1, these methods can not be thought of as solving a well-posed optimisation problem, and their repeated application may even be divergent as we observed. This can be seen as a major problem in the quest for performant, trustable machine learning for inverse problems, since it suggests that it will not be easy to adapt the convergence and stability results that we know from variational regularisation methods to learned iterative reconstruction methods.

For this purpose, we will have to impose extra structure on the learned reconstruction method. In Chapter 3, we took a step in this direction, ensuring the stability of learned denoisers for use in inverse problems solvers. It must be noted, however, that there remains room for improvement in this approach: provable stability comes at the cost of reconstruction quality, making it necessary to find a reasonable trade-off between stability guarantee and performance.

At the other extreme, we have the bilevel learning approach studied in Chapter 1, which comes with the guarantees of variational regularisation, but currently still lacks the flexibility and computational efficiency of the learned iterative reconstruction methods.

It goes without saying that the use of machine learning for inverse problems remains a highly active topic of research, and the above discussion suggests a two-pronged plan of attack to reach the goal of performant, trustable machine learning methods for inverse problems: either moving from bilevel learning in the direction of more flexibility and less structure, or moving from the learned iterative reconstruction methods in the direction of more structure and less flexibility. With this in mind, some interesting directions for future research are as follows:

Improved computations for bilevel learning problems

Although the bilevel learning approach to learning inverse problem solvers, as used in Chapter 1, has taken a back seat with the rise of deep learning approaches, it remains an interesting approach: the learned reconstruction methods are variational regularisation methods, with all of their desirable properties, and the methods are trained in an end-to-end manner to maximise reconstruction quality. By far the largest downside to these methods is the computational effort required. Recently, there has been some work showing that, when using a derivative-free optimisation method to solve a bilevel learning problem, large speedups can be achieved by using an inexact optimisation approach rather than always solving the lower level problems to high accuracy (Ehrhardt and Roberts, 2021). A disadvantage of the derivative-free optimisation approach compared to first-order gradient-based optimisation is that it is harder to scale to high dimensions (in the largest experiment in Ehrhardt and Roberts (2021) a 64-dimensional parameter vector is learned). Nonetheless, this work could be seen as inspiration to use an inexact gradient-based optimisation method, such as the method described in Sra (2012), to solve the bilevel learning problem. A basic requirement to apply such an inexact method is that we can bound the error in the computed gradient. Empirically, we have found that the method that we used in Chapter 1 to differentiate the lower level solution maps requires the

lower level problem to be solved to high accuracy to obtain a usable gradient. A promising alternative is to consider the use of an automatic differentiation (AD) approach; the AD community has previously studied the convergence of derivatives computed by automatically differentiating through a convergent iterative method (Gilbert, 1992; Griewank and Faure, 2002) and recently more refined results have been shown for problems similar to the bilevel learning problem (Mehmood and Ochs, 2020), essentially showing that the convergence rate of the gradient computed by AD is inherited from the convergence rate for the lower level solver. These results can be used to derive bounds on the gradients computed by AD, which in turn can be plugged into an inexact first-order method for the bilevel learning problem. We saw similar behaviour using our approach that invokes the implicit function theorem (recall Figure 1.3), but the AD approach has the additional advantage that we can compute the exact gradient of the lower level solver that we are using, so that we can be certain that we will avoid tripping up the upper level solver.

Parametrising functions with structural constraints

In Chapter 2, we saw that there is a vast amount of recent research into building neural networks that satisfy group equivariance properties. In the context of solving inverse problems there are additional structural constraints that we would like to enforce on the learnable functions that we use, as evidenced by these conditions repeatedly appearing as assumptions needed in theoretical analyses. In particular, it is often asked that a denoiser in a Plug-and-Play or regularisation by denoising algorithm has a symmetric Jacobian, and that the spectrum of the Jacobian is everywhere constrained to be nonnegative and bounded above by 1 (Chan, 2019; Reehorst and Schniter, 2019; Sreehari et al., 2016; Teodoro et al., 2019). This is for a good reason: by Moreau's characterisation of proximal operators (Moreau, 1965), these are necessary and sufficient conditions (assuming smoothness) for the denoiser to be the proximal operator of some convex functional. In fact, this characterisation can even be extended to proximal operators of nonconvex functionals by dropping the upper bound on the spectrum of the Jacobian (Gribonval and Nikolova, 2020). An interesting question that then arises, is whether we can exploit this characterisation to flexibly parametrise proximal operators, for instance by designing neural network architectures that have the aforementioned constraints built in. This would represent a step towards narrowing the gap between theory and practice. One simple observation that goes in this direction is the fact that, by Poincaré's lemma, the Jacobian symmetry condition is equivalent to the denoiser being the gradient of a functional. Adding the positivity constraint corresponds to asking that this underlying functional is

convex. It is well known that we can parametrise convex functionals $\phi : \mathbf{R}^d \to \mathbf{R}$ using neural networks with appropriately constrained weights (Amos et al., 2017); given such a neural network ϕ, $\Psi := \nabla\phi$ is a neural network that is the proximal operator of a (potentially nonconvex) functional and we can attempt to use this as a denoiser.

Definitions of performance measures

Let us note here the definitions of the two performance measures that we have used throughout the dissertation to evaluate image reconstruction quality:

- The peak signal to noise ratio (PSNR), defined for a ground truth signal $u^* \in \mathbf{R}^n$ and reconstruction $\hat{u} \in \mathbf{R}^n$ as

$$\mathrm{PSNR}(\hat{u}, u^*) = 10 \log_{10} \left(\frac{n \max_{1 \leqslant i \leqslant n} |u_i^*|^2}{\|u^* - \hat{u}\|^2} \right).$$

- The structural similarity index measure (SSIM) (Brunet et al., 2012), defined initially on small windows of images, $u^*, \hat{u} \in [0,1]^{w \times w}$ (with w odd) by

$$\mathrm{SSIM}(\hat{u}, u^*) = \frac{2\overline{\hat{u}} \cdot \overline{u^*} + c_1}{\overline{\hat{u}}^2 + \overline{u^*}^2 + c_1} \cdot \frac{2s_{\hat{u},u^*} + c_2}{s_{\hat{u}}^2 + s_{u^*}^2 + c_2}$$

  for small nonnegative constants c_1, c_2. In this formula, we have used the mean and variance statistics defined by

$$\overline{u} = \frac{1}{w^2} \sum_{1 \leqslant i,j \leqslant w} u_{i,j}^*, \quad s_{\hat{u},u^*} = \frac{1}{w^2} \sum_{1 \leqslant i,j \leqslant w} (\hat{u}_{i,j} - \overline{\hat{u}})(u_{i,j}^* - \overline{u^*}), \quad s_u = s_{u,u^*}.$$

  To obtain a performance measure for larger images $u^*, \hat{u} \in [0,1]^{n_1 \times n_2}$ with $n_1, n_2 \geqslant w$, we compute the SSIM on each of their subwindows and average:

$$\mathrm{SSIM}(\hat{u}, u^*) = \frac{1}{(n_1 + 1 - w)(n_2 + 1 - w)} \sum_{\substack{1 \leqslant i \leqslant n_1 + 1 - w \\ 1 \leqslant j \leqslant n_2 + 1 - w}} \mathrm{SSIM}([\hat{u}]_{i,j}, [u^*]_{i,j}),$$

  where $[u]_{i,j}$ is the window $(u_{k,l})_{i \leqslant k < i+w, j \leqslant l < j+w}$. We use the implementation included in scikit-image (Van der Walt et al., 2014), with the corresponding default parameter choices: $w = 7, c_1 = 10^{-4}, c_2 = 9 \cdot 10^{-4}$. It is worth noting that not all common implementations

of the SSIM use the same default parameter choices, and some may include weights in the computation of the subwindow statistics. As a result, reported SSIM values are not generally comparable across different works.

Both the PSNR and the SSIM have the property that higher values correspond to better reconstructions. To apply either of these performance measures to complex-valued images, we compute them on the absolute value images.

Bibliography

Aarle, W. v., Palenstijn, W. J., Cant, J., Janssens, E., Bleichrodt, F., Dabravolski, A., Beenhouwer, J. D., Batenburg, K. J., and Sijbers, J. (2016). Fast and flexible X-ray tomography using the ASTRA toolbox. *Optics Express*, 24(22):25129–25147.

Adcock, B., Hansen, A. C., Poon, C., and Roman, B. (2017). Breaking the coherence barrier: a new theory for compressed sensing. *Forum of Mathematics, Sigma*, 5:e4.

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

Adler, J. and Öktem, O. (2018). Learned Primal-dual Reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332.

Aggarwal, H. K. and Jacob, M. (2020). J-MoDL: Joint Model-Based Deep Learning for Optimized Sampling and Reconstruction. *IEEE Journal of Selected Topics in Signal Processing*.

Akhtar, N. and Mian, A. (2018). Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430.

Amos, B., Xu, L., and Kolter, J. Z. (2017). Input convex neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 august 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 146–155. PMLR.

Antun, V., Renna, F., Poon, C., Adcock, B., and Hansen, A. C. (2020). On instabilities of deep learning in image reconstruction and the potential costs of AI. *Proceedings of the National Academy of Sciences*, 117(48):30088–30095.

Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. *arXiv: 1701.07875*.

Armato III, S. G., McLennan, G., Bidaut, L., McNitt-Gray, M. F., Meyer, C. R., Reeves, A. P., and Clarke, L. P. (2015). Data from LIDC-IDRI. *The Cancer Imaging Archive*, 10.

Armato III, S. G., McLennan, G., Bidaut, L., McNitt-Gray, M. F., Meyer, C. R., Reeves, A. P., Zhao, B., Aberle, D. R., Henschke, C. I., Hoffman, E. A., Kazerooni, E. A., MacMahon, H., Beek, E. J. R. v., Yankelevitz, D., Biancardi, A. M., Bland, P. H., Brown, M. S., Engelmann, R. M., Laderach, G. E., Max, D., Pais, R. C., Qing, D. P.-Y., Roberts, R. Y., Smith, A. R., Starkey, A., Batra, P., Caligiuri, P., Farooqi, A., Gladish, G. W., Jude, C. M., Munden, R. F., Petkovska,

I., Quint, L. E., Schwartz, L. H., Sundaram, B., Dodd, L. E., Fenimore, C., Gur, D., Petrick, N., Freymann, J., Kirby, J., Hughes, B., Casteele, A. V., Gupte, S., Sallam, M., Heath, M. D., Kuhn, M. H., Dharaiya, E., Burns, R., Fryd, D. S., Salganicoff, M., Anand, V., Shreter, U., Vastagh, S., Croft, B. Y., and Clarke, L. P. (2011). 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.

Bauschke, H. H. and Combettes, P. L. (2011). *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer.

Beck, A. and Teboulle, M. (2009). A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

Bekkers, E. J., Lafarge, M. W., Veta, M., Eppenhof, K. A. J., Pluim, J. P. W., and Duits, R. (2018). Roto-Translation Covariant Convolutional Networks for Medical Image Analysis. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, pages 440–448.

Benning, M., Brune, C., Burger, M., and Müller, J. (2013). Higher-Order TV Methods—Enhancement via Bregman Iteration. *Journal of Scientific Computing*, 54(2):269–310.

Benning, M., Gladden, L., Holland, D., Schönlieb, C.-B., and Valkonen, T. (2014). Phase reconstruction from velocity-encoded MRI measurements - A survey of sparsity-promoting variational approaches. *Journal of Magnetic Resonance*, 238:26–43.

Boyer, C., Bigot, J., and Weiss, P. (2019). Compressed sensing with structured sparsity and structured acquisition. *Applied and Computational Harmonic Analysis*, 46(2):312–350.

Boyer, C., Chauffert, N., Ciuciu, P., Kahn, J., and Weiss, P. (2016). On the generation of sampling schemes for magnetic resonance imaging. *SIAM Journal on Imaging Sciences*, 9(4):2039–2072.

Bredies, K. and Holler, M. (2014). Regularization of linear inverse problems with total generalized variation. *Journal of Inverse and Ill-posed Problems*, 22:871 – 913.

Bredies, K., Kunisch, K., and Pock, T. (2010). Total Generalized Variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526.

Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. *arXiv:2005.14165*.

Broyden, C. G. (1970). The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90.

Bruck, R. E. (1977). On the weak convergence of an ergodic iteration for the solution of variational inequalities for monotone operators in Hilbert space. *Journal of Mathematical Analysis and Applications*, 61(1):159–164.

Brunet, D., Vrscay, E. R., and Wang, Z. (2012). On the Mathematical Properties of the Structural Similarity Index. *IEEE Transactions on Image Processing*, 21(4):1488–1499.

Burger, M. and Osher, S. (2004). Convergence rates of convex variational regularization. *Inverse Problems. An International Journal on the Theory and Practice of Inverse Problems, Inverse Methods and Computerized Inversion of Data*, 20(5):1411–1421.

Burrage, K. and Butcher, J. C. (1979). Stability criteria for implicit Runge–Kutta methods. *SIAM Journal on Numerical Analysis*, 16(1):46–57.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.

Calatroni, L., Chung, C., Reyes, J. C. D. L., Schönlieb, C.-B., and Valkonen, T. (2015). Bilevel approaches for learning of variational imaging models. *arXiv:1505.02120*.

Candès, E. J. and Donoho, D. (1999). Curvelets: A surprisingly effective nonadaptive representation for objects with edges. Technical Report 1999-28, Stanford University.

Candès, E. J., Romberg, J. K., and Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223.

Celledoni, E., Ehrhardt, M. J., Etmann, C., McLachlan, R. I., Owren, B., Schönlieb, C.-B., and Sherry, F. (2021a). Structure-preserving deep learning. *European Journal of Applied Mathematics*, pages 1–49.

Celledoni, E., Ehrhardt, M. J., Etmann, C., Owren, B., Schönlieb, C.-B., and Sherry, F. (2021b). Equivariant neural networks for inverse problems. *Inverse Problems*. https://doi.org/10.1088/1361-6420/ac104f.

Chambolle, A. (2004). An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20:89–97.

Chambolle, A. and Pock, T. (2011). A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145.

Chan, S. H. (2016). Algorithm-induced prior for image restoration. *arXiv:1602.00715*.

Chan, S. H. (2019). Performance analysis of plug-and-play ADMM: A graph signal processing perspective. *IEEE Transactions on Computational Imaging*, 5(2):274–286.

Chan, S. H., Wang, X., and Elgendy, O. A. (2016). Plug-and-play ADMM for image restoration: Fixed point convergence and applications. *arXiv:1605.01710*.

Chang, B., Meng, L., Haber, E., Ruthotto, L., Begert, D., and Holtham, E. (2018). Reversible Architectures for Arbitrarily Deep Residual Neural Networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 2811–2818. AAAI Press.

Chauffert, N., Ciuciu, P., Kahn, J., and Weiss, P. (2014). Variable density sampling with continuous trajectories. *SIAM Journal on Imaging Sciences*, 7(4):1962–1992.

Chauffert, N., Ciuciu, P., and Weiss, P. (2013). Variable density compressed sensing in MRI. Theoretical vs heuristic sampling strategies. In *2013 IEEE ISBI*, pages 298–301. IEEE.

Chaux, C., Combettes, P. L., Pesquet, J.-C., and Wajs, V. R. (2007). A variational formulation for frame-based inverse problems. *Inverse Problems. An International Journal on the Theory and Practice of Inverse Problems, Inverse Methods and Computerized Inversion of Data*, 23(4):1495–1518.

Chavent, G. and Kunisch, K. (1997). Regularization of linear least squares problems by total bounded variation. *ESAIM: Control, Optimisation and Calculus of Variations*, 2:359–376.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6572–6583.

Chen, Y. (2014). *Learning fast and effective image restoration models*. PhD thesis, Graz University of Technology.

Chen, Y., Pock, T., and Bischof, H. (2014). Learning l1-based analysis and synthesis sparsity priors using bi-level optimization. *arXiv:1401.4105*.

Cocosco, C. A., Kollokian, V., Kwan, R. K.-S., Pike, G. B., and Evans, A. C. (1997). BrainWeb: Online interface to a 3D MRI simulated brain database. *NEUROIMAGE*, 5:425.

Cohen, T. S., Geiger, M., and Weiler, M. (2019). A General Theory of Equivariant CNNs on Homogeneous Spaces. In *Advances in Neural Information Processing Systems*, volume 32, pages 9145–9156.

Cohen, T. S. and Welling, M. (2016). Group Equivariant Convolutional Networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 2990–2999.

Combettes, P. L. and Pesquet, J.-C. (2020). Deep Neural Network Structures Solving Variational Inequalities. *Set-Valued and Variational Analysis*, 28(3):491–518.

Cotter, F. and McLaughlin, S. (2019). fbcotter/pytorch_wavelets: Zenodo Release.

Coulon, O., Alexander, D. C., and Arridge, S. (2004). Diffusion tensor magnetic resonance image regularization. *Medical Image Analysis*, 8(1):47–67.

Dahlquist, G. and Jeltsch, R. (1979). Generalized disks of contractivity for explicit and implicit Runge-Kutta methods. Technical report, CM-P00069451.

De los Reyes, J. C. and Schönlieb, C.-B. (2013). Image denoising: Learning the noise model via nonsmooth PDE-constrained optimization. *Inverse Problems & Imaging*, 7(4):1183–1214.

De los Reyes, J. C., Schönlieb, C.-B., and Valkonen, T. (2015). The structure of optimal parameters for image restoration problems. *arXiv:1505.01953*.

De los Reyes, J. C., Schönlieb, C.-B., and Valkonen, T. (2017). Bilevel parameter learning for higher-order total variation regularisation models. *Journal of Mathematical Imaging and Vision*, 57(1):1–25.

Dempe, S. and Zemkoho, A. B. (2011). The generalized Mangasarian-Fromowitz constraint qualification and optimality conditions for bilevel programs. *Journal of Optimization Theory and Applications*, 148(1):46–68.

Dieleman, S., De Fauw, J., and Kavukcuoglu, K. (2016). Exploiting Cyclic Symmetry in Convolutional Neural Networks. In Balcan, M.-F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, pages 1889–1898.

Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306.

Ehrhardt, M. J. and Betcke, M. M. (2016). Multicontrast MRI reconstruction with structure-guided total variation. *SIAM Journal on Imaging Sciences*, 9(3):1084–1106.

Ehrhardt, M. J. and Roberts, L. (2021). Inexact Derivative-Free Optimization for Bilevel Learning. *Journal of Mathematical Imaging and Vision*, 63(5):580–600.

Engl, H. W., Hanke, M., and Neubauer, A. (1996). *Regularization of inverse problems*, volume 375 of *Mathematics and Its Applications*. Kluwer Academic Publishers, Dordrecht.

Feng, L., Grimm, R., Block, K. T., Chandarana, H., Kim, S., Xu, J., Axel, L., Sodickson, D. K., and Otazo, R. (2014). Golden-angle radial sparse parallel MRI: Combination of compressed sensing, parallel imaging, and golden-angle radial sampling for fast and flexible dynamic volumetric MRI. *Magnetic Resonance in Medicine*, 72(3):707–717.

Finzi, M., Stanton, S., Izmailov, P., and Wilson, A. G. (2020). Generalizing Convolutional Neural Networks for Equivariance to Lie Groups on Arbitrary Continuous Data. *arXiv:2002.12880*.

Fletcher, R. (1970). A new approach to variable metric algorithms. *The Computer Journal*, 13(3):317–322.

Folland, G. B. (2015). *A course in abstract harmonic analysis*. CRC Press, Boca Raton, 2 edition.

Gilbert, J. C. (1992). Automatic differentiation and iterative processes. *Optimization Methods and Software*, 1(1):13–21.

Goldfarb, D. (1970). A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24:23–26.

Golub, G. H. and Vorst, H. A. v. d. (2000). Eigenvalue computation in the 20th century. *Journal of Computational and Applied Mathematics*, 123(1):35–65.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Gözcü, B., Mahabadi, R. K., Li, Y.-H., Ilicak, E., Cukur, T., Scarlett, J., and Cevher, V. (2018). Learning-based compressive MRI. *IEEE Transactions on Medical Imaging*, 37(6):1394–1406.

Gözcü, B., Sanchez, T., and Cevher, V. (2019). Rethinking sampling in parallel MRI: A data-driven approach. In *27th European Signal Processing Conference (EUSIPCO)*, pages 1–5.

Gribonval, R. and Nikolova, M. (2020). A characterization of proximity operators. *arXiv:1807.04014*.

Griewank, A. and Faure, C. (2002). Reduced Functions, Gradients and Hessians from Fixed-Point Iterations for State Equations. *Numerical Algorithms*, 30(2):113–139.

Guerquin-Kern, M., Van De Ville, D., Vonesch, C., Baritaux, J.-C., Pruessmann, K., and Unser, M. (2009). Wavelet-regularized reconstruction for rapid MRI. In *2009 IEEE ISBI*, pages 193–196. IEEE.

Guo, K. and Labate, D. (2007). Optimally sparse multidimensional representation using shearlets. *SIAM Journal on Mathematical Analysis*, 39(1):298–318.

Hadamard, J. (1902). Sur les problèmes aux derivées partielles et leur signification physique. *Princeton University Bulletin*, XIII(4):49–52.

Hairer, E. and Wanner, G. (1996). *Solving Ordinary Differential Equations II*, volume 14 of *Springer Series in Computational Mathematics*. Springer-Verlag Berlin Heidelberg.

Haldar, J. P. and Kim, D. (2019). OEDIPUS: An experiment design framework for sparsity-constrained MRI. *IEEE Transactions on Medical Imaging*.

Hall, B. (2015). *Lie groups, Lie algebras, and representations. An elementary introduction*, volume 222. Cham: Springer.

Hammernik, K., Klatzer, T., Kobler, E., Recht, M. P., Sodickson, D. K., Pock, T., and Knoll, F. (2018). Learning a variational network for reconstruction of accelerated MRI data. *Magnetic Resonance in Medicine*, 79(6):3055–3071.

Hansen, P. C. (2010). *Discrete inverse problems. Insight and algorithms*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM).

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

Hasannasab, M., Hertrich, J., Neumayer, S., Plonka, G., Setzer, S., and Steidl, G. (2020). Parseval Proximal Neural Networks. *Journal of Fourier Analysis and Applications*, 26(4).

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning. Data mining, inference, and prediction*. New York, NY: Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Hertrich, J., Neumayer, S., and Steidl, G. (2020). Convolutional Proximal Neural Networks and Plug-and-Play Algorithms. *arXiv:2011.02281*.

Hofmanninger, J., Prayer, F., Pan, J., Röhrich, S., Prosch, H., and Langs, G. (2020). Automatic lung segmentation in routine imaging is primarily a data diversity problem, not a methodology problem. *European Radiology Experimental*, 4(1):50.

Horesh, L., Haber, E., and Tenorio, L. (2010). Optimal experimental design for the large-scale nonlinear ill-posed problem of impedance imaging. In *Large-Scale inverse problems and quantification of uncertainty*, pages 273–290. John Wiley & Sons, Ltd. Section: 13 tex.eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470685853.ch13.

Hounsfield, G. N. (1973). Computerized transverse axial scanning (tomography): Part 1. Description of system. *British Journal of Radiology*, 46(552):1026–1022.

Hu, Y. and Jacob, M. (2012). Higher degree total variation (HDTV) regularization for image recovery. *IEEE Transactions on Image Processing*, 21(5):2559–2571.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101.

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

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Tunyasuvunakool, K., Ronneberger, O., Bates, R., Žídek, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Potapenko, A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Steinegger, M., Pacholska, M., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. (2020). High Accuracy Protein Structure Prediction Using Deep Learning. In *Fourteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstract Book)*.

Kelley, C. T. and Sachs, E. W. (1987). Quasi-newton methods and unconstrained optimal control problems. *SIAM Journal on Control and Optimization*, 25(6):1503–1516.

Kingma, D. P. and Ba, J. (2017). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*.

Knoll, F., Bredies, K., Pock, T., and Stollberger, R. (2011a). Second order total generalized variation (TGV) for MRI. *Magnetic Resonance in Medicine*, 65(2):480–491.

Knoll, F., Clason, C., Diwoky, C., and Stollberger, R. (2011b). Adapted random sampling patterns for accelerated MRI. *Magnetic Resonance Materials in Physics, Biology and Medicine*, 24(1):43–50.

Knoll, F., Zbontar, J., Sriram, A., Muckley, M. J., Bruno, M., Defazio, A., Parente, M., Geras, K. J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdzalv, M., Romero, A., Rabbat, M., Vincent, P., Pinkerton, J., Wang, D., Yakubova, N., Owens, E., Zitnick, C. L., Recht, M. P., Sodickson, D. K.,

and Lui, Y. W. (2020). 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(1):e190007.

Krahmer, F. and Ward, R. (2014). Stable and robust sampling strategies for compressive imaging. *IEEE Transactions on Image Processing*, 23(2):612–622.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105.

Kunisch, K. and Pock, T. (2013). A Bilevel Optimization Approach for Parameter Learning in Variational Models. *SIAM Journal on Imaging Sciences*, 6:938–983.

Lauterbur, P. C. (1973). Image Formation by Induced Local Interactions: Examples Employing Nuclear Magnetic Resonance. *Nature*, 242(5394):190–191.

Lazarus, C., Weiss, P., Chauffert, N., Mauconduit, F., El Gueddari, L., Destrieux, C., Zemmoura, I., Vignaud, A., and Ciuciu, P. (2019). SPARKLING: variable-density k-space filling curves for accelerated T_2^*-weighted MRI. *Magnetic Resonance in Medicine*, 81(6):3643–3661.

LeCun, Y. and Bengio, Y. (1998). Convolutional networks for images, speech, and time series. In *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

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

Li, Y.-H. and Cevher, V. (2016). Learning data triage: Linear decoding works for compressive MRI. In *2016 IEEE ICASSP*, pages 4034–4038. IEEE.

Lingala, S. G., Hu, Y., Dibella, E., and Jacob, M. (2011). Accelerated dynamic MRI exploiting sparsity and low-rank structure: k-t SLR. *IEEE Transactions on Medical Imaging*, 30(5):1042–1054.

Liu, J. and Saloner, D. (2014). Accelerated MRI with CIRcular Cartesian UnderSampling (CIRCUS): a variable density Cartesian sampling strategy for compressed sensing and parallel imaging. *Quantitative Imaging in Medicine and Surgery*, 4(1):57–67.

Lunz, S., Öktem, O., and Schönlieb, C.-B. (2018). Adversarial Regularizers in Inverse Problems. In *Advances in Neural Information Processing Systems*, volume 31, pages 8516–8525.

Lustig, M., Donoho, D., and Pauly, J. M. (2007a). Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195.

Lustig, M., Donoho, D., and Pauly, J. M. (2007b). Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195.

Mansfield, P. and Grannell, P. K. (1975). "Diffraction" and microscopy in solids and liquids by NMR. *Physical Review B*, 12(9):3618–3634.

Mehmood, S. and Ochs, P. (2020). Automatic differentiation of some first-order methods in parametric optimization. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 august 2020, online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of machine learning research*, pages 1584–1594. PMLR.

Meinhardt, T., Möller, M., Hazirbas, C., and Cremers, D. (2017). Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. *arXiv:1704.03488*.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *arXiv:1802.05957*.

Moore, E. H. (1920). On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26:394–395.

Moreau, J. J. (1962). Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 255:2897–2899.

Moreau, J. J. (1963). Propriétés des applications "prox". *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 256:1069–1071.

Moreau, J. J. (1965). Proximité et dualité dans un espace hilbertien. *Bulletin de la Société mathématique de France*, 93:273–299.

Mukherjee, S., Dittmer, S., Shumaylov, Z., Lunz, S., Öktem, O., and Schönlieb, C.-B. (2021). Learned convex regularizers for inverse problems. *arXiv:2008.02839*.

Nevanlinna, O. and Sipilä, A. H. (1974). A nonexistence theorem for explicit A-stable methods. *Mathematics of computation*, 28(128):1053–1056.

Nocedal, J. and Wright, S. J. (2006). *Numerical optimization*. New York, NY: Springer.

Osher, S., Burger, M., Goldfarb, D., Xu, J., and Yin, W. (2005). An iterative regularization method for total variation-based image restoration. *Multiscale Modeling and Simulation*, 4:460–489.

Palenstijn, W. J., Batenburg, K. J., and Sijbers, J. (2011). Performance improvements for iterative electron tomography reconstruction using graphics processing units (GPUs). *Journal of Structural Biology*, 176(2):250–253.

Papafitsoros, K. and Schönlieb, C.-B. (2014). A Combined First and Second Order Variational Approach for Image Reconstruction. *Journal of Mathematical Imaging and Vision*, 48(2):308–338.

Paquette, M., Merlet, S., Gilbert, G., Deriche, R., and Descoteaux, M. (2015). Comparison of sampling strategies and sparsifying transforms to improve compressed sensing diffusion spectrum imaging. *Magnetic Resonance in Medicine*, 73(1):401–416.

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

Passty, G. B. (1979). Ergodic convergence to a zero of the sum of monotone operators in Hilbert space. *Journal of Mathematical Analysis and Applications*, 72(2):383–390.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8026–8037.

Pejoski, S., Kafedziski, V., and Gleich, D. (2015). Compressed sensing MRI using discrete nonseparable shearlet transform and FISTA. *IEEE Signal Processing Letters*, 22(10):1566–1570.

Penrose, R. (1955). A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413.

Pesquet, J.-C., Repetti, A., Terris, M., and Wiaux, Y. (2020). Learning Maximally Monotone Operators for Image Recovery. *arXiv:2012.13247*.

Piccini, D., Littmann, A., Nielles-Vallespin, S., and Zenge, M. O. (2011). Spiral phyllotaxis: The natural way to construct a 3D radial trajectory in MRI. *Magnetic Resonance in Medicine*, 66(4):1049–1056.

Poon, C. (2016). On Cartesian line sampling with anisotropic total variation regularization. *arXiv:1602.02415*.

Putzky, P. and Welling, M. (2017). Recurrent Inference Machines for Solving Inverse Problems. *arXiv:1706.04008*.

Puy, G., Vandergheynst, P., and Wiaux, Y. (2011). On variable density compressive sampling. *IEEE Signal Processing Letters*, 18(10):595–598.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Ravishankar, S. and Bresler, Y. (2011a). Adaptive sampling design for compressed sensing MRI. In *2011 Annual International Conference of the IEEE EMBS*, pages 3751–3755. IEEE.

Ravishankar, S. and Bresler, Y. (2011b). MR image reconstruction from highly undersampled k-Space data by dictionary learning. *IEEE Transactions on Medical Imaging*, 30(5):1028–1041.

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

Romano, Y., Elad, M., and Milanfar, P. (2017). The Little Engine That Could: Regularization by Denoising (RED). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, volume 9351 of *Lecture notes in computer science*, pages 234–241. Springer.

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

Ruthotto, L. and Haber, E. (2020). Deep Neural Networks Motivated by Partial Differential Equations. *Journal of Mathematical Imaging and Vision*, 62(3):352–364.

Ryu, E. K., Liu, J., Wang, S., Chen, X., Wang, Z., and Yin, W. (2019). Plug-and-Play Methods Provably Converge with Properly Trained Denoisers. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5546–5557. PMLR.

Samuel, K. G. G. and Tappen, M. F. (2009). Learning optimized MAP estimates in continuously-valued MRF models. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 477–484.

Sanz Serna, J. and Zygalakis, K. C. (2020). Contractivity of Runge–Kutta Methods for Convex Gradient Systems. *SIAM Journal on Numerical Analysis*, 58(4):2079–2092.

Scherzer, O. (2007). Denoising with higher order derivatives of bounded variation and an application to parameter estimation. *Computing*, 60:1–27.

Seeger, M., Nickisch, H., Pohmann, R., and Schölkopf, B. (2009). Optimization of k-space trajectories for compressed sensing by Bayesian experimental design. *Magnetic Resonance in Medicine*, 63(1):116–126.

Shanno, D. F. (1971). Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24:647–656.

Sherry, F., Benning, M., De los Reyes, J. C., Graves, M. J., Maierhofer, G., Williams, G., Schönlieb, C.-B., and Ehrhardt, M. J. (2020). Learning the Sampling Pattern for MRI. *IEEE Transactions on Medical Imaging*, 39(12):4310–4321.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144.

Sodickson, D. K., Feng, L., Knoll, F., Cloos, M., Ben-Eliezer, N., Axel, L., Chandarana, H., Block, T., and Otazo, R. (2015). The rapid imaging renaissance: sparser samples, denser dimensions, and glimmerings of a grand unified tomography. In *Proceedings of SPIE*, volume 9417, page 94170G. International Society for Optics and Photonics.

Sommerhoff, H., Kolb, A., and Moeller, M. (2019). Energy Dissipation with Plug-and-Play Priors. In *NeurIPS 2019 Workshop on Solving Inverse Problems with Deep Networks*.

Sosnovik, I., Szmaja, M., and Smeulders, A. (2019). Scale-Equivariant Steerable Networks. *arXiv:1910.11093*.

Sra, S. (2012). Scalable nonconvex inexact proximal splitting. In Bartlett, P. L., editor, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 539–547.

Sreehari, S., Venkatakrishnan, S. V., Wohlberg, B., Buzzard, G. T., Drummy, L. F., Simmons, J. P., and Bouman, C. A. (2016). Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2(4):408–423.

Sutton, R. S. (2019). The Bitter Lesson. Available at http://www.incompleteideas.net/IncIdeas/BitterLesson.html.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9.

Tenorio, L., Lucero, C., Ball, V., and Horesh, L. (2013). Experimental design in the context of Tikhonov regularized inverse problems. *Statistical Modelling*, 13(5-6):481–507. tex.eprint: https://doi.org/10.1177/1471082X13494613.

Teodoro, A. M., Bioucas-Dias, J. M., and Figueiredo, M. A. T. (2019). A convergent image fusion algorithm using scene-adapted gaussian-mixture-based denoising. *IEEE Transactions on Image Processing*, 28(1):451–463.

Teschl, G. (2012). *Ordinary differential equations and dynamical systems*, volume 140. Providence, RI: American Mathematical Society (AMS).

Trémoulhéac, B., Dikaios, N., Atkinson, D., and Arridge, S. R. (2014). Dynamic MR image reconstruction-separation from undersampled (k, t)-Space via low-rank plus sparse prior. *IEEE Transactions on Medical Imaging*, 33(8):1689–1701.

Usman, M. and Batchelor, P. G. (2009). Optimized sampling patterns for practical compressed MRI. *SampTA'09*.

Vaiter, S., Deledalle, C., Fadili, J., Peyré, G., and Dossal, C. (2017). The degrees of freedom of partly smooth regularizers. *Annals of the Institute of Statistical Mathematics*, 69(4):791–832.

van Aarle, W., Palenstijn, W. J., De Beenhouwer, J., Altantzis, T., Bals, S., Batenburg, K. J., and Sijbers, J. (2015). The ASTRA Toolbox: A platform for advanced algorithm development in electron tomography. *Ultramicroscopy*, 157:35–47.

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T. (2014). scikit-image: image processing in Python. *PeerJ*, 2:e453.

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

Virmaux, A. and Scaman, K. (2018). Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, I., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and

Contributors (2020). SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272.

Weiler, M. and Cesa, G. (2019). General E(2)-Equivariant Steerable CNNs. In *Advances in Neural Information Processing Systems*, volume 32, pages 14334–14345.

Weiler, M., Geiger, M., Welling, M., Boomsma, W., and Cohen, T. S. (2018a). 3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data. In *Advances in Neural Information Processing Systems*, volume 32, pages 10381–10392.

Weiler, M., Hamprecht, F. A., and Storath, M. (2018b). Learning Steerable Filters for Rotation Equivariant CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 849–858.

Weiss, T., Vedula, S., Senouf, O., Michailovich, O., Zibulevsky, M., and Bronstein, A. (2020). Joint Learning of Cartesian under Sampling and Reconstruction for Accelerated MRI. In *2020 IEEE ICASSP*, pages 8653–8657. IEEE.

Welling, M. (2019). Do we still need models or just more data and compute? Available at https://staff.fnwi.uva.nl/m.welling/wp-content/uploads/Model-versus-Data-AI-1.pdf.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. (2017). Harmonic Networks: Deep Translation and Rotation Equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5028–5037.

Worrall, D. E. and Welling, M. (2019). Deep Scale-spaces: Equivariance Over Scale. *arXiv:1905.11697*.

Zbontar, J., Knoll, F., Sriram, A., Murrell, T., Huang, Z., Muckley, M. J., Defazio, A., Stern, R., Johnson, P., Bruno, M., Parente, M., Geras, K. J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdzal, M., Romero, A., Rabbat, M., Vincent, P., Yakubova, N., Pinkerton, J., Wang, D., Owens, E., Zitnick, C. L., Recht, M. P., Sodickson, D. K., and Lui, Y. W. (2019). fastMRI: An Open Dataset and Benchmarks for Accelerated MRI. *arXiv:1811.08839*.

Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. (2017a). Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155.

Zhang, K., Zuo, W., Gu, S., and Zhang, L. (2017b). Learning deep CNN denoiser prior for image restoration. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2808–2817.

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

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1997). Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 23(4):550–560.