Data-driven Methods in Inverse Problems
Data-driven Methods in Inverse Problems

JONAS ADLER

Doctoral Thesis
KTH – Royal Institute of Technology
School of Engineering Sciences
Department of Mathematics
Stockholm, Sweden 2019
Akademisk avhandling som med tillstånd av Kungliga Tekniska Högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen, Torsdag den 31 Oktober 2019 klockan 14:00 i sal F3, Lindstedtsvägen 26, Kungliga Tekniska Högskolan, Stockholm.

© Jonas Adler, 2019

Print: Universitetsservice US AB, Stockholm, 2019
Abstract

In this thesis on data-driven methods in inverse problems we introduce several new methods to solve inverse problems using recent advancements in machine learning and specifically deep learning. The main goal has been to develop practically applicable methods, scalable to medical applications and with the ability to handle all the complexities associated with them.

In total, the thesis contains six papers. Some of them are focused on more theoretical questions such as characterizing the optimal solutions of reconstruction schemes or extending current methods to new domains, while others have focused on practical applicability. A significant portion of the papers also aim to bringing knowledge from the machine learning community into the imaging community, with considerable effort spent on translating many of the concepts. The papers have been published in a range of venues: machine learning, medical imaging and inverse problems.

The first two papers contribute to a class of methods now called learned iterative reconstruction where we introduce two ways of combining classical model driven reconstruction methods with deep neural networks. The next two papers look forward, aiming to address the question of "what do we want?" by proposing two very different but novel loss functions for training neural networks in inverse problems. The final papers delve into the statistical side, one gives a generalization of a class of deep generative models to Banach spaces while the next introduces two ways in which such methods can be used to perform Bayesian inversion at scale.

Keywords: Inverse Problems, Machine Learning, Tomography
Sammanfattning

Den här avhandlingen om datadrivna metoder för inversa problem introducerar flera nya metoder för att lösa inversa problem med hjälp av nya framsteg inom maskininlärning och specifikt djupinlärning. Målet har varit att utveckla praktiskt applicerbara metoder som skal till kliniska datastorlekar och som kan hantera de komplexiteter som är associerade med dem.


De första två artiklarna bidrar till en klass av metoder som nu kallas *inlärda iterativa rekonstruktion* där vi introducerar två nya sätt att kombinera klassiska modelldrivna rekonstruktionsmetoder med djupa neuronnät. De nästa två artiklarna blickar framåt och försöker ange om "vad vill vi?" genom att föreslå två väldigt olika men nya målfunktioner för att tränna neurala nät för inversa problem. De sista artiklarna gräver djupare i den statistiska sidan, ett ger en generalisering av en klass av djupa generativa modeller till Banachrum medan det sista introducerar två nya sätt som data-drivna metoder kan användas för storskalig Bayesisk inversion.
Acknowledgments

Doing a PhD is often said to be a lonely endeavour, but loneliness cannot describe my experience and the list of people who have made this thesis possible is far too long. This is but a brief overview.

I started my research journey during my undergrad and I have many to thank. Not least to the physics chapter at KTH who gave me friendship, curiosity and love that I hope will last for a lifetime. I’m also indebted to the research group at St. Jude who dared to take on a first year student and got me hooked on research.

I also extend my gratitude to Elekta who took me in and let me roam free and grow. Special thanks to the phenomenal Research and Physics group, to Markus for your inexhaustible work on imaging, to Jens for our endless and inspiring discussions, to Håkan for your support and finally to Jonas G and Anna who made all of this possible.

Thanks to everyone who has been part of making the imaging group at KTH so amazing, to Gustav, Axel and Sebastian B for taking my crazy ideas and turning them into something sensible, mostly without questioning my sanity. To Olivier for always sharing your wisdom, David and Mamo for making sure we’re actually doing something useful and to Jevgenija who made me appreciate the importance of free food. But of course the greatest credit goes to you Ozan, who has given me more guidance and support than any PhD student could wish for.

I would also like to thank everyone in the mathematics department at KTH. Special thanks to Anders S who’s experience helped keep us on track and to Johan without whom I never would have understood the strength of weakness. And to my teachers, Henrik S, Henrik H, Anders F, Elias, Krister, Hans, Mattias and everyone else who have taught me about math and more.

Thanks also to the imaging community at large and to all the great and not so great ideas that we have discussed over N drinks. To Simon, Carola and Peter for your mentoring, to Andreas for all of our insightful discussions, to Sebastian L who had to put up with me saving money on hotels, to Holger who kept me in check and to Roy who made me appreciate it all. Thanks also to everyone at CWI for all the great meetings and software, without you we would all be looking at much more boring images.

I also extend my thanks to everyone at Google and DeepMind and especially to the Science team which has welcomed me to an exiting environment where I can
keep building on many of the insights I have gained from working on this thesis.

Finally, there are many that have helped keeping me sane and happy during these years. A huge thank you to all of my friends, you are simply too many to list, but special thanks to Micke who has made sure I stay in shape both physically and mentally. Thanks to my family, Lena, Anders and Sofie for being the bedrock upon which my life is built and to my extended family, the Adlers, Holmérs and Morfar for always being there and keeping me down to earth.

But of course my greatest gratefulness lies with Ally, who has put up with all my ups and downs, with late evenings and travel and who always believed in me more than anyone could reasonably expect, both at home and at work. I love you so much.

London, October 2019

Jonas Adler
Table of Contents

Abstract iii
Acknowledgments v
Table of Contents vii

Part I: Introduction
1  Image Reconstruction 3
2  Machine Learning 15
3  Wasserstein Distances 25

Part II: Research Papers
“If we knew what we were doing, it wouldn’t be called research.”
Part I: Introduction
1. Image Reconstruction

In several areas of science and technology there is a need to study the interior of objects without opening them up. A prime example comes from medicine where we are often interested in probing the interior of the patient in order to determine e.g. where a tumour is. For much of human history invasive probing using surgery was the only feasible alternative, but it is dangerous to the patient and can only be performed in limited areas.

A revolution occurred with the introduction of X-ray imaging by Wilhelm Röntgen whose seminal article "On a new kind of ray: A preliminary communication" was submitted on December 28, 1895 [70]. For the first time in human history, we could look inside the human body without cutting it open. The field of radiology quickly took off, with the method used in clinical practice less than two weeks later (legal requirements seem to have been more relaxed during this time) and in just the year of 1896, Science published 23 articles on X-rays.

In X-ray radiography a beam of X-ray photons is directed at the object of study and we look at its shadow. Dense materials attenuate X-rays more strongly than lighter materials, meaning that the photons interact more often with them, thus reducing the intensity of the beam as it passes through the object. The photons fluence is then measured on the other side of the patient and a image is formed where dark regions correspond to rays of photons that have passed through heavier materials such as metal or bone. The process, along with the first medical X-ray is shown in figs. 1 and 2.

However, simple X-ray radiography did not fully solve the problem of looking inside humans. While X-ray radiography allows us to see the interior of the patient, the images we see are in fact aggregate information with each point in the radiograph being influenced by the attenuation throughout the whole object. This causes a problem since the depth of different structures cannot be determined and they may end up superimposed in the image, see fig. 3. Thus, one can not find the exact three dimensional position of e.g. a tumour from a planar radiograph.

The first X-ray based method that allowed actual tomography (lit. to write slices), e.g. computing how the body looks in a specific point in the interior rather than some aggregate value, was focal plane tomography. In this technique, illustrated
1. **Image Reconstruction**

Figure 1: Illustration of the working procedure in X-ray radiography. X-rays are emitted from the source (top) and pass through the object (middle) before being measured by the detector (bottom).

Figure 2: The first ever medical radiograph.

Figure 3: Examples of objects with the same radiograph.
1.1. Image Formation in CT

Figure 4: Illustration of the working procedure in focal plane tomography. Each set of lines with the same colour intersect the focal plane (gray) in one unique point. The radiodensity in this point is hence accumulated in the detector.

In fig. 4, the X-ray source and detector are both moved relative to the patient. They are moved such that the projection of a specific plane (the focal plane) does not move in the detector, but all other planes do. Hence, the only objects that will appear sharp in the image are those in the focal plane. This is however a non-perfect technique. First, we are not getting the exact values in the slice, but rather the values of adjacent slices are merely smoothed out. Further, the modality is only able to image one slice at a time and the radiation dose is quite high.

Thus, there was a need for further innovation and this came with the invention of the Computed Tomography (CT) in the early 70s, roughly contemporarily with Magnetic Resonance Imaging (MRI). In CT, radiographs of the patient are taken from a multitude of directions, see fig. 5. Each of these will represent a piece of aggregate information about the patients interior, but as proven in the celebrated article of Johann Radon [65] we can use the combined information in the radiographs to solve for the attenuation everywhere inside.

1.1 Image Formation in CT

We shall now give a more formal description of how data is acquired and how volumetric images are reconstructed from the measurement data, and we will focus on CT as an example.

The image formation process in CT is governed by the transport of individual photons through the object [75]. The fluence, photons in some state per time unit,
1. Image Reconstruction

Figure 5: Illustration of the working procedure in X-ray tomography. The source and the detector rotate jointly around the patient. Each colour corresponds to one X-ray image, these are then combined to find the volume.

of this process is described by the radiative transport equation, a integro-differential equation which can be solved to determine the image that the detector would measure in the absence of noise given information about the X-ray source and the attenuation of the patient. Our task in image reconstruction in CT is to determine the spatial distribution of the attenuation coefficient given images measured by the detector.

In the standard practical treatment of CT one assumes that, under some simplifying assumptions and after sufficient pre-processing of the measurement data, each pixel in the detector measures a line integral of the attenuation coefficient of the patient. If we denote the attenuation (units: reciprocal length) by $x$, the measured data by $y$ and we let $\ell$ denote a line from the source to the detector then they are related through

$$y(\ell) = \int_{\ell} x ds$$

In inverse problems, this is typically simplified by gathering the physics into a forward operator $A$, allowing us to rewrite the above equation as

$$y = Ax$$

where the forward operator in eq. (1.1.1) is called the Radon transform in 2d and the ray transform in 3d.

This formulation is useful since we can develop solution methods without having to deal with all the intricate details related to the physics. In this thesis, the main
application has indeed been X-ray based tomography but the methodology has been developed with a much more general scope in mind. This is clearly interesting in case the methods should be applied to some other modality, for example if a company suddenly decides to use MRI imaging instead of CT.

Normally there is also some noise associated with the imaging setup. In X-ray imaging the main source of noise is typically quantum noise caused by the quantization of the X-rays as a finite number of photons. This noise approximately follows a Poisson distribution, which in the low-noise limit is often approximated with a Gaussian distribution whose variance is proportional to the square root of the incoming signal.

1.2 Inverse Problems

The field of inverse problems is, in a sense, the study of solving eq. (1.1.2) under various assumptions [56]. Our practical objective is to take some measurement data $y$ and reconstruct the parameters $x$ that gave rise to them, but there has also been intense theoretical interest in inverse problems chiefly focusing on proving existence, uniqueness and stability of solutions.

While computed tomography is an archetypical example, there is a wide range of problems that can be phrased as inverse problems. Most notably is perhaps image reconstruction in medicine such as MRI, Single-Photon Emission Computed Tomography (SPECT) and Positron Emission Tomography (PET) but inverse problems are also widely studied in image reconstruction for seismic imaging and for parameter estimation in PDEs among many others. Further, several problems in image processing such as denoising, super-resolution and deblurring can also be seen as inverse problems. While they may seem simple in comparison to e.g. computed tomography, they share several mathematical characteristics and can be used to highlight some properties of interest in inverse problems and as a testing ground for algorithms.

Inverse problems are typically classified in terms of their ill-posedness which can be roughly related to how hard they are to solve. The mathematical definition of ill-posedness is due to Hadamard [23] who defined a problem as well-posed if

- A unique solution exists
- The solution depends continuously on the measurement

while a problem is ill-posed if either of these fail to apply.

For the case of X-ray tomography in two dimensions, existence and uniqueness was settled more than 100 years ago by Johan Radon [65] while the three dimensional setting was solved much later, around the time when clinical 3D tomographs started being used [82]. Some corner cases are still subject to active research, e.g. for

2Any similarity with an actual company based in Stockholm, Sweden is purely coincidental.
acquisitions of moving objects [24]. Nonetheless, uniqueness typically holds in most clinical CT scanners.

Most practical tomography setups are designed to ensure that uniqueness holds, but it is often the case that practical constraints force us to abandon this. For example, cone-beam CT reconstruction is not unique which gives rise to a characteristic artefact. Likewise many modern MRI scanners under-sampled in order to speed up the image acquisition. From a mathematical standpoint this missing information is not too interesting, if nothing was measured then we simply have to guess, inpaint, using our prior knowledge.

On the other hand continuity of the solution never holds in CT. To see why, we can use the Singular Value Decomposition (SVD) of $A$. In the 2D case the SVD was explicitly computed in [50] but for a general compact forward operator $A$ it has the form

$$Ax = \sum_{n=0}^{\infty} \sigma_n u_n \langle v_n, x \rangle$$

(1.2.1)

where $u_n$ is a orthonormal basis for the domain, $v_n$ a orthonormal basis for the range and $\sigma_n \in \mathbb{R}$ are the singular values. Low $n$ correspond to low frequencies, while high $n$ correspond to high frequencies. For the ray transform the singular values $\sigma_n$ decay as $O(n^{-0.5})$. Since the pseudo-inverse is given by

$$A^\dagger y = \sum_{n=0}^{\infty} \sigma_n^{-1} v_n \langle u_n, y \rangle$$

(1.2.2)

we note that it is not a bounded operator, hence not continuous. To expand upon this let $e$ be some noise, then we can look at how the reconstruction error behaves at various frequencies by computing

$$\left\langle v_n, A^\dagger (Ax + e) - x \right\rangle = \left\langle v_n, \sum_{n=0}^{\infty} \sigma_n^{-1} v_n \left( \sum_{n=0}^{\infty} \sigma_n u_n \langle v_n, x \rangle + e \right) - x \right\rangle = \sigma_n^{-1} \langle u_n, e \rangle$$

Which shows, perhaps unsurprisingly, that the low frequency content in the error corresponds to low frequency content in the noise. But it also shows that this is scaled by $\sigma_n^{-1} = O(n^{0.5})$ and hence the high frequency noise gets amplified. This implies that recovering the low frequency components is typically relatively easy, while recovering the high frequency components such as edges is problematic.

This situation is not uncommon and occurs in several other inverse problems with most practical imaging modalities having polynomial decay, some even exhibit exponential decay. This has very interesting consequences, it means that we actually measure everything we need but that for sufficiently high frequencies noise will be increasingly dominant. Finding an appropriate way of balancing the noise and signal is hence very important.
In general for these ill-posed inverse problems we can thus not apply the pseudo-inverse directly, but we need to somehow deal with the fact that these higher frequencies become increasingly hard to recover. An extensive literature on this topic has evolved over more than 50 years, starting with the seminal works of Tikhonov [81].

Broadly speaking, there are two main schools of thought in the theory of regularization of inverse problems. The first and historically most widely studied is the functional-analytic approach where we design inversion schemes by investigating their functional properties, for example stable recovery under certain assumptions. An interesting class of such methods is SVD based approaches. Some examples include SVD truncation which only uses singular values larger than a certain value $\sigma_0$

$$A_{\text{truncate}}^t y = \sum_{n: \sigma_n > \sigma_0} \sigma_n^{-1} v_n \langle u_n, y \rangle$$ (1.2.3)

and SVD based Tikhonov regularization which modifies the singular values of the inverse to make sure that they are bounded and eventually go to zero

$$A_{\text{Tikhonov}}^t y = \sum_{n=0}^{\infty} \frac{\sigma_n}{\sigma_n^2 + \alpha^2} v_n \langle u_n, y \rangle$$ (1.2.4)

Both of these methods can be shown to recover the lower frequencies without adding high frequency noise. For example, the truncation approach gives

$$\langle v_n, A_{\text{truncate}}^t (Ax + e) - x \rangle = \begin{cases} \sigma_n^{-1} \langle u_n, e \rangle & \text{if } \sigma_n > \sigma_0 \\ \langle v_n, x \rangle & \text{else} \end{cases}$$

where we have a trade-off between the first term (the variance) which becomes small for large $\sigma_0$ and the second term (the bias) which becomes small for small $\sigma_0$. Since $\sigma_n \to 0$ there is no universally optimal $\sigma_0$, a large value will give a high bias even at low frequencies, while a small value will give high noise.

A related class of methods that is specialized for ray transform inversion is so called filtered back-projection methods. As noted above the inverse ray transform increases high frequency components, and this is typically counteracted by using a so called apodization filter [56]. Thanks to a property of the ray transform, the Fourier slice theorem, this filtering can be done in a computationally efficient manner by filtering the input data before applying the pseudo-inverse. Due to its relative ease of implementation, good results and fast runtime these methods have been the standard in commercial CT for more than 30 years.

However, starting around 2004 the development of so called compressive sensing started to gain traction thanks to ground-breaking theoretical advances in sampling theory that promised exponential improvements in sample complexity, something that would allow image reconstruction from far less data. The initial promises were perhaps overstated [60] but these methods have recently started being used in commercial scanners.
1. Image Reconstruction

While compressive sensing based imaging techniques were historically developed in a functional analytic or signal processing framework, it fits beautifully into the Bayesian framework which I will now introduce and which will be useful when interpreting these methods in the light of machine learning.

Bayesian Inverse Problems

The statistical interpretation of inverse problems started gaining traction even before the rise of machine learning, largely because it gives us a very flexible framework to construct and interpret regularization schemes, and because it allows us to reason about uncertainties in a formal manner [35, 79].

Here we assume that not only is the measured data a random (due to noise), but that this randomness gives rise to an uncertainty in the reconstruction. In this setting, our belief (some might say knowledge) about the reconstruction given our measurement data can be represented by the posterior distribution. That is, the probability of a given reconstruction given what we know, both from before (prior information) and from the measured data. The posterior distribution is typically denoted $P(x = x \mid y = y)$, often shortened to $P(x \mid y)$ and denoting the probability (density) if the random variable $x$ being equal to $x$ given that the random variable $y$ is equal to $y$.

Bayes’ theorem provides a convenient way of expressing this probability as a function of more easily computed individual components

$$P(x \mid y) = \frac{P(y \mid x)P(x)}{P(y)}$$

where the denominator is a normalization which we could in theory compute using the law of total probability

$$P(y) = \int P(y \mid x)dP(x) \quad (1.2.5)$$

Using Bayes law we note that the posterior is uniquely characterized by two components. The data likelihood $P(y \mid x)$ gives the probability of measuring a certain data given a reconstruction and this is typically well known in terms of the physics of the problem. For example for the Radon transform with Poisson noise the data likelihood is well approximated using a normal distribution

$$P(y \mid x) \propto e^{-\frac{1}{2} \| A x - y \|_\Lambda^{-1/2}}$$

where the covariance can be approximated by $\Lambda = \text{diag}(y_0 e^{-A x})$ where $y_0$ is the number of photons hitting each pixel in the case where no object is present.

On the other hand, the prior $P(x)$ represents our belief in how the reconstruction should look like. Since it represents a belief, the picking a prior is subjective with no objectively correct choice. So how do we then pick a prior? We can start by noting
that the prior should reflect our belief about how the object we are imaging looks like and in the best of worlds, our prior should incorporate everything we know about the object but not more. We might for example know that we’re imaging a human male at 50 years of age, but we might not know if he has a tumour or not. However, in practice this form of prior information is very hard to encode mathematically and we have traditionally had make do with much weaker prior assumptions, typically using some measure of the smoothness of the image.

Once we have derived the data likelihood and decided on a prior, the posterior distribution is in principle determined. However, the whole posterior distribution is typically hard to work with and interpret, not to mention computationally infeasible, and hence one typically resorts to computing various estimators that characterize the posterior.

There is a large set of estimators to chose from, and preferably we would pick an estimator that is relevant to whatever application we’re interested in. This is typically not the case. Rather, estimators are generally selected with the condition that they are efficiently computable, which has significantly limited the options available to classical approaches.

By far the most popular estimator used in imaging is the Maximum a posteriori (MAP) estimate which is obtained by solving

$$\text{MAP}(y) = \arg \max_{x \in X} P(x \mid y)$$

This estimator is useful because the optimization problem in question can be easily solved in a range of cases. In particular if we consider the logarithm (which is an increasing function) we find

$$\arg \max_{x \in X} P(x \mid y) = \arg \max_{x \in X} \log P(x \mid y)$$

$$= \arg \max_{x \in X} \log \frac{P(y \mid x)P(x)}{P(y)}$$

$$= \arg \max_{x \in X} (\log P(y \mid x) + \log P(x) - \log P(y))$$

Hence, not only can we ignore the $P(y)$ term which would otherwise have involved a nasty marginalization in eq. (1.2.5) but if $-\log P(y \mid x)$ and $-\log P(x)$ are convex the problem can be reduced to a convex optimization problem. Convex problems are nice since every local optimum is a global optimum and there is a wealth of algorithms for solving them reasonably fast [13, 15].

Example 1.2.1 (Gaussian prior). To illustrate the above, assume that the forward model is linear from $\mathbb{R}^k$ to $\mathbb{R}^n$ and that we have a Gaussian prior with mean $\mu$ and
covariance $\Sigma$. Then the prior likelihood reads

$$
P(x) \propto e^{-\frac{1}{2} \|x - \mu\|^2_{\Sigma^{-1}/2}}
$$

and

$$
-\log P(x) = \frac{1}{2} \|x - \mu\|^2_{\Sigma^{-1}/2} + C
$$

where $C$ is a constant that is irrelevant to the minimizer. Further we assume that the noise is Gaussian with mean zero, implying that the data has mean $A(x)$, and covariance $\Lambda$,

$$
P(y \mid x) \propto e^{-\frac{1}{2} \|Ax - y\|^2_{\Lambda^{-1}/2}}
$$

and

$$
-\log P(y \mid x) = \frac{1}{2} \|Ax - y\|^2_{\Lambda^{-1}/2} + C'
$$

Where once again $C'$ is independent of $x$. Combining this we find

$$
\arg\max_{x \in X} P(x \mid y) = \arg\min_{x \in X} - (\log P(y \mid x) + \log P(x))
$$

$$
= \arg\min_{x \in X} \frac{1}{2} \left( \|Ax - y\|^2_{\Lambda^{-1}/2} + \|x - \mu\|^2_{\Sigma^{-1}/2} \right)
$$

MAP estimates for Gaussian priors are especially nice to work with since they have a closed form solution. In particular, the above can be minimized by setting its gradient to zero

$$
0 = \nabla_x P(x \mid y) = A^T \Lambda^{-1} (Ax - y) + \Sigma^{-1} (x - \mu)
$$

$$
= (A^T \Lambda^{-1} A + \Sigma^{-1}) x - A^T \Lambda^{-1} y - \Sigma^{-1} \mu
$$

which has a closed form solution

$$
x = (A^T \Lambda^{-1} A + \Sigma^{-1})^{-1} (A^T \Lambda^{-1} y + \Sigma^{-1} \mu) \quad (1.2.6)
$$

Even in the case where explicitly computing this solution is intractable, simple iterative solutions such as quasi-Newton schemes [49] typically converge very quickly to a solution. Not also that Tikhonov regularization becomes a special case of MAP estimation under Gaussian priors.

This class of regularizers are very widely used due to their ease of use, and they often appear as building blocks in more advanced regularization schemes [76]. However, the restriction to Gaussian priors is often quite severe and only relatively simple priors can be represented.

Example 1.2.2 (Sparsity inducing priors). While Gaussian priors are nice to work with, they often fail to capture many of the properties that we expect from our reconstructions. In particular the linearity of the reconstruction operator eq. (1.2.6) implies that the solution operator decreases all features equally when compared to the pseudo-inverse, hence if we want to reduce all high frequency noise by 90%
we must also decrease the high frequency signal by as much. This often leads to solutions that are overly smooth with no clear edges. To solve this problem, sparsity inducing priors have been investigated following the seminal paper by [73] in 1992. In short we assume that there is some basis in which the signal is sparse. For example, we could assume that the edges, e.g. the image’s gradient, is sparse. We then try to find the reconstruction that is as sparse as possible in this basis.

There are two main formulations, analysis and synthesis. In the analysis framework we assume that there is an analysis operator \( D \) such that \( Dx \) is sparse. In the synthesis framework we assume that \( x = B\xi \) where \( \xi \) is sparse. In case these operators are invertible, the formulations are equivalent but if they are not there are subtle differences. We’ll focus on the analysis formulation since it is the most widely used in practice with Total Variation (TV) regularization being the most common use case.

The most straightforward prior that uses this formulation is probably

\[
- \log P(x) = \lambda \| Dx \|_0 + C
\]

where \( \| \cdot \|_0 \) indicates the 0-”norm”, the measure of the support of \( Dx \), e.g. the number of non-zero elements if \( X \) is finite dimensional. Under this prior we assume that sparser \( x \) are more probable than less sparse ones. However, a major problem with this formulation is that it is NP-hard to find the corresponding MAP solution.

A solution to this is to make a convex relaxation, to pick a prior which is similar to the above prior but where the MAP solution can be more easily found. The prior of choice has turned out to use the 1-norm in place of the 0-norm:

\[
- \log P(x) = \lambda \| Dx \|_1 + C
\]

It might (should?) not be intuitively obvious to the reader as to why this would give sparse solutions. In particular many non-sparse solutions, e.g. \( x = \varepsilon x_0 \) where \( \varepsilon \) is a tiny number and \( x_0 \) is any signal of bounded (1-)norm, would be considered very likely under this prior, yet is typically not sparse.

The reason that this prior gives rise to sparse solutions using the MAP estimator lies in the interaction of the prior with the data-likelihood. To illustrate this we can consider denoising \((\mathcal{A} = I)\) with Gaussian white noise and \( D = I \) which gives the MAP estimate

\[
\arg \min_x \left[ \frac{1}{2} \| x - y \|^2 + \lambda \| x \|_1 \right]
\]

so the data term is quadratic while the regularizer is (locally) linear. This means that where \( x \) is small the sparsity inducing 1-norm term will dominate and where \( x \) is large the quadratic term dominates. Hence values that would have been small without the regularizer get pushed down to zero, while larger values are mostly untouched. By carefully tuning the trade-off parameter \( \lambda \) sparsity can then be achieved.
There has also been some work in Bayesian inverse problems aiming to use non-MAP estimators. A notable favourite is the conditional mean

$$\mathbb{E}[x \mid y] = \int xdP(x \mid y)$$  \hspace{1cm} (1.2.8)

which has several favourable properties including almost trivial existence and stability for most finite dimensional inverse problems, something that is not nearly as obvious for the MAP estimator. However, while the conditional mean and MAP estimators happen to coincide in the Gaussian case (example 1.2.1) which makes computing the conditional mean easy, this does not hold in general and for other priors computing the conditional mean is typically very computationally costly using traditional methods, often involving the use of Markov chain Monte Carlo (MCMC) sampling [51].
2. Machine Learning

While classical Bayesian methods for inverse problems have many upsides, one of their foremost downsides is that the prior has to be specified by hand. For some problems where the natural prior has a simple structure this might work very well but more often than not the natural choice of prior, e.g. that we’re imaging a human being, is intractable since we cannot specify what this means in purely mathematical terms. Machine learning methods solve this problem by at least partially specifying the prior in terms of examples, e.g. “it looks like this”.

The general setting is as follows, we assume that there is some (unknown) probability distribution of images, $x$, that we seek to reconstruct given measurement data from a distribution of measurement data $y$. In order to learn something about these distributions we draw random samples from at least one of these and use these examples to guide us in constructing a better reconstruction operator, e.g. to pick some parameters.

By now there exists a wide range of methods for machine learning in inverse problems, and they can be broadly classified according to what form of data used for training them. There are two main types of data, examples of reconstructions $x_i \sim x$, and examples of measured data $y_j \in y$. These may be combined in various ways but by far the most popular are unsupervised and supervised methods. In addition to these there exists a wealth of other less commonly used methods [47, 95] that we omit for brevity.

A major topic in classical machine learning is that of generalization, e.g. does what we have learned from a few examples apply to the whole distribution. This has been well studied in the case of more simple classical methods [11, 78] but is currently not well understood for more recent deep learning based methods.

2.1 Unsupervised Learning

We’ll start by giving a overview of unsupervised methods for image reconstruction. These “learned prior” methods use only examples of reconstructions to train a machine learning model that tries to approximate some property of the prior $P(x = x)$. 
The seemingly most straightforward method is to directly estimate the prior density from the training data and the most simple approach is to simply use the empirical distribution

$$P_{\text{empirical}}(x) = \frac{1}{n} \sum_{i=1}^{n} \delta(x - x_i).$$

but this distribution has support only on the training data and hence does not generalize very well to unseen data, assigning zero probability to almost all of them. An extension that solves this is histogramming, e.g. separating $X$ into bins and approximating the density in each bin by the number of points it contains. However since we need to cover the space with bins and we also want a reasonable number of data-points per bin in order to get a reasonable estimate it turns out that the optimal bin size is $O(n^{-1/(d+2)})$ where $d$ is the dimension of $X$. Even at modest dimensions, this implies that we need impossibly many samples to achieve bin sizes that are small enough to resolve $X$ to any reasonable accuracy.

Even more advanced approximations, such as Kernel Density Estimation (KDE)

$$P_{\text{KDE}}(x) = \frac{1}{hn} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

where $K : \mathbb{R} \rightarrow \mathbb{R}$ is a kernel (a non-negative function) and $h$ is the width of the kernel, are remarkably weak. It can be shown that the optimal width $h$ is $O(n^{-1/(d+4)})$. This is clearly significantly better than histogramming in low dimensions but the advantage quickly diminishes. It can be shown under weak assumptions that this convergence rate is asymptotically optimal for all non-parametric methods [86]. Hence, it is quite clear that classical non-parametric density estimation is not a feasible approach for learning a prior in high dimensional inverse problems.

Thus we need to turn to parametric methods, e.g. we need to prescribe a model for the prior and then select some of its parameters using training data. The standard approach in statistics is to select some parametrized prior $P_\theta(x)$ and use e.g. a maximum likelihood estimate to select the parameters

$$\theta^* = \arg \max_\theta \text{log } P_\theta(x_1, \ldots, x_n)$$

which can be vastly simplified in the setting of independent training data to

$$\theta^* = \arg \max_\theta \left[ \prod_{i=1}^{n} P_\theta(x_i) \right] = \arg \min_\theta \left[ -\sum_{i=1}^{n} \log P_\theta(x_i) \right] \quad (2.1.1)$$

A very wide range of models can be built upon this idea, and we’ll mention some of the most important.

The first unsupervised learning technique that gained widespread acceptance in the inverse problems community was (sparse) dictionary learning . While there are
several related concepts all called the same thing, the main idea is to learn a basis in which the training data \( x_i \) are sparse, e.g. can be represented by a small number of elements. This is typically done by learning the analysis or synthesis operator in eq. (1.2.7) and one way of doing so in the analysis setting [72] is to select the dictionary where the training data is the most sparse (on average), e.g. let

\[
- \log P_D(x) = \|Dx\|_0
\]

and find the best \( D \) according to eq. (2.1.1). However, due to noise and other effects, it is often impossible to find a basis where the signals are jointly sparse and one often looks at relaxed versions of the above, by allowing a bit of slack. This is commonly further relaxed to make the problem convex by replacing the 0-semi norm with the 1-norm. Several extensions to this general setup have been investigated in the literature, notably using different potential functions instead of the 0 or 1-norms.

However, these methods are typically parametrized by a small number of parameters and we need more representative power in order to approximate the type of distributions encountered in practice. A methods that tries to solve this is the Field of Experts (FoE) model [71] which uses a more richly parametrized prior,

\[
\log P_{\theta}(x) = \sum_i f_{\theta_i,1}(J_{\theta_i,2}x)
\] (2.1.2)

where \( f_{\theta_i,1} : X \to \mathbb{R} \) are functionals with some learnable parameters and \( J_{\theta_i,2} : X \to X \) are linear operators with some learnable parameters, e.g. convolutions parametrized by their kernels.

Still, these methods puts significant restrictions on the class of priors that can be learned to those that can be represented by one or more dictionaries, which clearly does not cover the use-cases we were looking for, e.g. the prior of assuming that we’re imaging a human. To solve this issue of insufficient representative power, researchers have been increasingly looking towards using neural networks.

Very briefly, deep neural networks can be seen as multi-level function approximators where multiple affine functions (typically convolutions in imaging) are composed with pointwise non-linearities. The networks are heavily overparametrized with a number of parameters typically in the millions, some state of the art networks for natural language modelling use billions of parameters [18, 64, 89]. The parameters are selected using some form of gradient descent, which requires differentiation of these multi-level models using automatic differentiation and a vast number of very high quality open source software packages [1, 62] and high performance hardware [34] have been developed to accommodate this\(^1\). For a more thorough introduction see e.g. [21, 44, 77].

\(^1\)A significant part of the work during this thesis was spent developing a Python framework for inverse problems, Operator Discretization Library (ODL) [3] which has similar functionality but with inverse problems in mind. The framework has been made available as open source software on GitHub at https://github.com/odlgroup/odl.
These neural networks can be proven to have arbitrarily large representative power if the networks are large enough [30]. Hence, proper application of neural networks allows us to scale with big data. During the last few years, contemporaneously with the writing of this thesis, there has been a veritable explosion of methods that do just that.

Some of these techniques aim to find a closed form expression for the prior distribution (or e.g. its logarithm) in the form of a neural network. The first such techniques were deep autoregressive models [58] that use the chain rule for probabilities to describe the probability of a image as a product of the probabilities of all pixels conditioned on all earlier pixels. Assume $X$ is finite dimensional, e.g. $x = [x_1, x_2, \ldots, x_d]$ then we may write

$$P(x) = P(x_1, \ldots, x_{d-1})P(x_d | x_1, \ldots, x_{d-1}) = \prod_{i=1}^{d} P(x_i | x_{<i})$$

Hence we can decompose the probability into a sequence of $d$ 1-dimensional conditional probabilities so in practice all we need to do is to train a neural network that takes "all previous pixels" as input and outputs a probability distribution over the next pixel. Since this is a 1-dimensional distribution it can be modelled explicitly using e.g. histogramming. This has been used extensively in sequence modelling where it gives very good results [59] whereas its application to images is less limited. There are several reasons or this but perhaps most important is that the sequence becomes very long for high resolution images and the method is quadratic in the number of pixels, although there are some solutions to this [74] the methods still tend to become unwieldy for large images.

Another method for finding closed form priors is to use invertible models [39, 68]. Perhaps surprisingly, there are several methods that yield neural networks with closed form [16, 20, 32] or easily computed [9] inverses. If $\Lambda_\theta : X \rightarrow X$ is such an invertible neural network with Jacobian determinant $\det \partial \Lambda(x)$ then it represents a change of variables. If further $z$ is some $X$ valued random variable with known probability density $P(z = z)$ then we can define a prior according to

$$\log P_\theta(x = x) = \log P(\Lambda_\theta(x) = z) + \log |\det \partial \Lambda(x)|$$

We can then select the parameters according to eq. (2.1.1). Once trained, we then have a model which allows us to compute prior probabilities in closed form. The method can also be used to generate samples according to $P_\theta$ by generating a sample $z$ from $x$ and computing $\Lambda_\theta^{-1}(z)$. The method has so far only been sparsely applied to inverse problems, but there are some promising starts [6].

Another class of techniques exploit that neural networks, and in particular CNN, are phenomenal at denoising images [93]. Hence it is quite possible to train a neural network that is a next to optimal denoiser given some distribution of data [48], which can then be used to approximate some property of the prior.

One such ingenious technique called REgularization by Denoising (RED) [4, 57, 69] uses an Tweedie’s formula [19] which shows that if $D_\epsilon$ is the optimal denoiser
2.2 Supervised Learning

(minimizing the expected squared L2 error) given white noise of magnitude $\epsilon$ then we have

$$\lim_{\epsilon \to 0} \frac{D_\epsilon(x) - x}{\epsilon^2} = \nabla \log P(x)$$

This can be used in any gradient based optimization scheme to find an approximate MAP solution by training a denoiser at some small but fixed noise-level $\epsilon$ and then using it to approximate the gradient of the logarithm of the prior. For example, using simple gradient descent with step length $\gamma$ sufficiently small, we get a scheme that converges to an approximation of the MAP solution

$$x^{i+1} = x^i - \gamma \left( \nabla \log P(y|x^i) + \frac{D_\epsilon(x^i) - x^i}{\epsilon^2} \right)$$

The method can even be combined with Langevin dynamics to sample from the posterior distribution in order to compute e.g. the conditional mean [57] or the variance.

Another technique uses the proximal operator, defined [13] for convex functions $f : X \to \mathbb{R}$ by

$$\text{prox}_f(x) = \arg \min_{\tilde{x} \in X} \left[ f(\tilde{x}) + \frac{1}{2} \|x - \tilde{x}\|^2 \right]$$

It is useful since (among other things), it can be proven that the forward-backward scheme

$$x^{i+1} = \text{prox}_f(x^i - \gamma \nabla \log P(y|x^i))$$

where $\gamma \in \mathbb{R}$ is a step size, converges to a maximizer of $\log P(y|x) + f(x)$. The Plug and Play (PnP) [83] methods exploits that if $D : X \to X$ is a denoiser satisfying some conditions then it is the proximal operator of some function $f_D$ and hence the scheme

$$x^{i+1} = D(x^i - \gamma \nabla \log P(y|x^i))$$

will converge to a maximizer of $\log P(y|x) + f_D(x)$ where $f_D$ can be interpreted as the log-prior of some (unknown) distribution.

In addition to these methods there is a wealth of other methods using deep learning for learning a prior, including Approximate Message Passing (AMP) [55] and adversarial regularizers [52].

2.2 Supervised Learning

While unsupervised methods are useful since they can be trained using only samples of high quality reconstructions, they typically end up with some form of optimization problem that has to be solved to find e.g. the MAP estimate. This has some major downsides, first solving this optimization problem is often quite time consuming and second it is in general highly non-convex so finding a global optimum is very hard, and this has to be done each time the method is applied. Further, if one
wants to find some other estimator, e.g. the conditional mean, then even more
computationally intensive methods such as Langevin dynamics need to be used.

Supervised methods use matched pairs \((x_i, y_i) \sim (x, y)\) and learn a model for
mapping data to signal, thus removing the need of solving any optimization problem
during evaluation. This class of methods has been the main object of study in this
thesis, and I’ll try to give an overview of methods that were introduced before and
during the course of this thesis while aiming to highlight the contributions of this
thesis and how it relates to the other methods.

Supervised learning methods for inverse problems are typically parametric, e.g.
we have a reconstruction operator \(A_\theta^\dagger : Y \to X\) parametrized by \(\theta\) and we want
to select the parameters such that the reconstructions we obtain are as good as
possible. To do this, we need to define what a good reconstruction is \[10\]. We
typically do this by defining a loss function \(\ell : X \times X \to \mathbb{R}\) which measures how
good a reconstruction is by comparing it to the ground truth solution. The most
common choice is the squared \(L_2\) distance \(\ell(x_1, x_2) = \|x_1 - x_2\|^2\), and other typical
choices would be distances such as the \(L_p\) distances. However, more advanced
choices have been explored in the literature and the topic of one paper in this thesis
is on using Wasserstein metrics, but more on that later.

Using the loss function we thus have a way of defining if a single reconstruction
is good, but we need to push this to a way of defining how good a reconstruction
operator is. The standard method in statistical decision theory and machine learning
is to do this by computing some form of aggregate, e.g. the worst case loss, sometimes
called minimax, or the average, expected, loss. The expected loss is particularly
useful in large scale machine-learning applications since computing the loss on a
single sample is an unbiased estimator of the total loss, hence we can approximate
the total loss by the average loss of a small number of samples. Because of this
we can scale to extremely large data sizes and this scaling property has made
the expected loss become so common in Machine Learning (ML) applications that
alternatives are rarely investigated. We too shall progress along this line, but note
that this is an explicit choice we have made.

Combining all of this, we see that the optimal choice of parameters would be
given by minimizing the expected risk

\[
\theta^* = \arg\min_{\theta} \mathbb{E}\left[\ell(A_\theta^\dagger(y), x)\right]
\]

but of course we do not have access to the full random variables \(y, x\), we only
have access to a finite number of training samples. However, as mentioned before
approximating the expected loss by a small number of samples is an unbiased
estimator, and hence we can approximate the true expected loss by the empirical
risk

\[
\frac{1}{n} \sum_{i=1}^{n} \ell(A_\theta^\dagger(y_i), x_i)
\]

We can hence pick our parameters by instead minimizing the empirical risk, a
process called Empirical Risk Minimization (ERM) and which gives rise to a choice
2.2. Supervised Learning

of parameters

\[ \theta^{\text{ERM}} = \arg \min_{\theta} \left[ \frac{1}{n} \sum_{i=1}^{n} \ell(A_{\theta}^{\dagger}(y_i), x_i) \right] \]  

(2.2.1)

We note that solving this minimization problem exactly or even to within some a priori known error bound is typically impossible and a significant literature has been devoted to efficiently solving it approximately. For neural networks and deep learning in particular, the most successful methods by far make use of variants of Stochastic Gradient Descent (SGD) [38].

With respect to using ERM, it is clear that for non-degenerate cases \( \theta^* \neq \theta^{\text{ERM}} \) and hence \( A_{\theta^*}^{\dagger} \neq A_{\theta^{\text{ERM}}}^{\dagger} \) which implies that the reconstruction operator learned by ERM does not work optimally on unseen data, it over-fits to the training data. The study of this discrepancy has been a central tenet of machine learning for many years [78]. Several steps towards its solution have been attempted, but in general there is a large discrepancy between the theory and some experiments [91] which both show that deep learning based methods could in theory memorize the whole training dataset, giving an unbounded generalization error, and practical experience that show that the networks over-fit by only a little [43, 66]. A theoretical explanation of these successes is still considered somewhat of a holy grail in the community.

Another related direction is so called out of distribution stability, where we are interested in knowing how well a method trained for some distribution (e.g. women) works when applied to some other distribution (e.g. men). Here there are some theoretical results [45] and some learned reconstruction methods have been analysed w.r.t. this by other authors [12, 40]. However, just like the generalization gap there his a huge discrepancy between a weak theory and the remarkable practical results.

The final step in training these supervised learning schemes is to specify how the class of operators \( A_{\theta}^{\dagger} \) should look like. In particular, we need our reconstruction operators to map input data \( y \in Y \) to reconstructions \( A_{\theta}^{\dagger}(y) \in X \). We can start by considering an important class of problems, namely where \( Y = X \). In case \( X \) is a space of images, as is often the case, this is called an image processing problem, and we’re solving an inverse problem such as denoising or de-blurring. For these problems picking an architecture is especially simple since there is no need to map between possibly very different spaces and even the identity operator could be considered a reconstruction operator in this case, it even turns out to be a good initial guess [29, 92]. However while these inverse problems are interesting, they mostly appear in computational photography whereas we are interested in medicine, and in medicine we almost always have \( X \neq Y \). In this case, we need to be much more careful about how to pick our class of reconstruction operators and it turns out that many techniques become problem dependent to deal with the various difficulties associated with their respective forward operator. We’ll now give a broad overview of techniques that have been developed for this setting.
Bi-level Optimization

Perhaps the most straightforward and well understood application of supervised machine learning to solve inverse problems is bi-level optimization [14, 67]. The idea here is quite simple: most closed form priors have at least one free parameter that the user has to tune by hand in order to fully specify the prior, e.g. we can write our prior on the form $P_{\theta}(x)$ where $\theta$ is a parameter. For example with a Gaussian prior the covariance and mean has to be selected by hand, e.g. $\theta = (\mu, \Sigma)$. If we select an estimator, e.g. the MAP estimator we get a parametrized reconstruction operator of the form

$$A^\dagger_{\theta}(y) = \arg\min_x [-\log P(y | x) - \log P_{\theta}(x)]$$

where the parameters can be selected using ERM as in eq. (2.2.1). Since this type of learning problem is relatively simple, there has been quite extensive theoretical progress in proving e.g. existence of an optimal parameter choice and also continuity under some regularity assumptions. However, since this class of method deals with classical regularization methods, we are limited by the expressibility of e.g. 2-parameter families which generally cannot adequately represent the complex distributions appearing in nature.

Fully Learned Reconstruction

Perhaps a orthogonal approach to the above is to consider the reconstruction operator as an operator $\mathbb{R}^n \to \mathbb{R}^m$ and represent it using a fully connected neural network. This has been attempted repeatedly [7, 61, 94] but in general the approach fails to scale to real data since the fully connected layers need at least $nm$ parameters which becomes prohibitively large for even moderately sized problems, and hence the biggest case where this has been applied to date are low resolution 2d images. Due to these scaling issues a fully learned approach has little hope of scaling to higher dimensions without significant modifications.

Learned Post-Processing

A class of methods that has been very successful in practice is to use a classical inversion technique to convert the problem from a hard $Y \to X$ problem into a much easier $X \to X$ problem. In particular, let $A^\dagger : Y \to X$ be any classical reconstruction technique, then the learning problem $A^\dagger(y) \to x$ is a image processing problem and the wealth of methods from image processing can be applied. This has been done by several authors [26, 33, 36, 90] to great effect and also ties in very well with earlier efforts from e.g. industrial actors in constructing efficient and accurate $A^\dagger$. However, a lingering worry is that since $A^\dagger(y)$ is used as input then any information lost by the initial reconstruction cannot be recovered. This is also supported by several experiments in this thesis which indicate that these results give slightly worse results than more model based approaches.
Learned Iterative Reconstruction

Given what we’ve observed with the aforementioned approaches it seems that the sweet spot in terms of inductive bias might lie somewhere in between the more data-heavy approaches such as fully learned and post-processing, and the more model heavy bi-level approaches.

Learned iterative approaches do just that. They incorporate things that we know about the physics, such as the forward operators $\mathcal{A}$ and its adjoint $\mathcal{A}^*$ as components in a deep neural network. Since these operators map between the domain and range they can be used to solve the problem of specifying an architecture that maps between these two spaces and since we otherwise build the network like a deep neural network we get access to practically unbounded representative power.

There are multiple ways to do this, but the most popular has been to unroll a model driven optimization scheme to a fixed number of iterations, typically about ten, and then replace some component such as computing the gradient or proximal of the log-prior with a not-too-deep neural network. The full unrolled scheme can then be considered as a rather large neural network and trained as such. A major practical complication here is that we need to be able to interface the forward operator with whatever framework is used to model the neural network.

The development of these learned iterative schemes started with the publishing of ADMM-net around 2016 [80]. The method replaces several components in a unrolled Alternating Direction Method of Multipliers (ADMM) optimization solver with deep neural networks and trains the resulting network end to end. They presented strong results for a simplified form of MRI reconstruction (subsampled Fourier inversion).

Approximately at the same time, Variational Networks were developed [25, 42]. The method can be related to the Field of Experts and bi-level optimization schemes in that a prior is learned, but the prior is selected to be optimal given a fixed number of iterations, which vastly simplifies the training (and evaluation) as compared to bi-level optimization which uses "infinitely many" iterations. The method can also be trained with a different prior in each iteration which seems to give improved results. It was originally applied to MRI reconstruction, but has since been applied to other modalities including CT and ultrasound imaging [41, 85].

Finally, from the image processing community came Recurrent Inference Machines [63]. These methods unroll gradient descent but rather than learning simply a prior they learn the whole update operator. Hence they are slightly more machine learning heavy than the aforementioned approaches. We were inspired by the simplicity of this approach and the first article of this thesis builds upon the idea by including more inductive bias and by showing that the method is applicable to large scale inverse problems, in our case CT. The second article of this thesis further expands upon this idea by instead unrolling a primal-dual optimization scheme. The field of related methods have since exploded various architectural improvements [27, 28, 53] with applications to a wide range of problems [31, 87, 88] and with some theoretical analysis [54].
3. Wasserstein Distances

The field of statistical distances deals with computing distances between probability measures and has had a resurgence due to recent trends in using big data. However, computing distances between measures is seemingly unrelated to image reconstruction, but it turns out to be very useful in several applications where these distances arise naturally as loss functions and half of the papers in this thesis has made heavy use of a particular subclass of these distances: the Wasserstein distances.

For simplicity we’ll restrict the introduction to probability measures. Our main problem can be stated as such: we want to define a statistical distance, a distance between two \( X \)-valued random variables. Several such metrics have been studied in the literature, perhaps most notably the Kullback-Leibler divergence between probability measures \( \mu_1 \) and \( \mu_2 \)

\[
D_{KL}(\mu_1, \mu_2) = \int \log \left( \frac{d\mu_1}{d\mu_2} \right) d\mu_1
\]

Statistical distances can be broadly classified by their strength as measured by how well they can measure convergence. A statistical distance \( D_1 \) can intuitively be said to be stronger than the distance \( D_2 \) if convergence under \( D_1 \) guarantees convergence under \( D_2 \).

While being a strong measure is seemingly a good property, especially when training neural networks and we want to guarantee that they are correct, there are several cases where one would want a weaker metric. Consider for example the following canonical case [8]: Let \( D \) be a statistical distance and \( \delta_x \) be the Dirac measure centred on \( x \), then one would in many cases wish to have \( \lim_{x \to 0} d(\delta_x, \delta_0) = 0 \). This is however not the case for strong metrics, for the Kullback-Leibler (KL)-divergence we in fact have \( D_{KL}(\delta_x, \delta_0) = \infty \) for \( x \neq 0 \), while the somewhat weaker TV-distance has \( D_{TV}(\delta_x, \delta_0) = 1 \) under the same conditions. There is hence a need for weaker distances, and the Wasserstein distance is one such weak distance.

The Wasserstein distance is often motivated in terms of transportation: how much does it cost to "move" one distribution to another. Consider for example the case of moving Gaussian shaped heaps of sand centred on \( \pm x_0 \) to another. There are several ways to do this, one could for example move sand from point \( x \) to \(-x\),
3. Wasserstein Distances

Figure 1: Different distributions (transport maps) with the same marginals. The first transport map to the left is given by $T(x) = -x$ and is highly non-optimal while the second transport map $T(x) = x + 2x_0$ is optimal. To the right are two distributions that the Monge formulation cannot deal with but which the Wasserstein formulation can, including the average of the worst case and optimal case (far right).

which would seem terribly inefficient as compared to moving sand from $x$ to $x + 2x_0$. To formalize this, we follow Monge and define a transport map $T : X \rightarrow X$ which describes how each point is moved and we define the cost of moving as the distance moved weighted by the amount moved. The transport distance between the random variables $x_1$ and $x_2$ is hence given by

$$\min_T \mathbb{E} d(T(x_1), x_1)$$

where the minimization is taken over all transport maps such that $T(x_1) \overset{d}{=} x_2$ and $d : X \times X \rightarrow \mathbb{R}$ is a distance (between elements in $X$, not probability distributions). While this gives a very nice formalization of the problem of moving one distribution into another, it turns out that it does not work work very well in many cases. Consider for example moving a singular Dirac-delta heap into a two (half as large) Dirac-deltas, the formulation requires that each point is moved to a unique other point and hence we cannot achieve the goal even approximately. It is also asymmetric since the opposite is in fact possible. Another major problem with the formulation is that the optimization problem over transport maps is highly non-linear and hard to solve.

The Wasserstein metric [84] can be seen as a convex relaxation of the above optimization problem. Instead of moving each point to some unique point we allow each point to be spread out over all other points. More formally, instead of optimizing over a set of functions $X \rightarrow X$ with a requirement on the distribution of $T(x)$ we instead optimize over all joint distributions on $X \times X$ with a requirement on both conditionals. We hence seek to find the $X \times X$ valued distribution $\pi^*$ that minimizes

$$\min_\pi \mathbb{E}_{(x_1, x_2) \sim \pi} d(x_1, x_2)$$

where the infimum is taken over all $X \times X$ values random variables with marginals $x_1, x_2$. 26
In many cases this relaxed formulation gives rise to the same solution as the Monge formulation, but it is much easier to solve. The optimization problem is in fact simply a linear optimization problem with linear constraints, although rather high dimension since we minimize over distributions on the product space. A further interesting part of this formulation is that it gives rise to a statistical distance just like the KL divergence but where we have $D_{\text{Wasserstein}}(\delta_x, \delta_0) = x$, thus giving us the convergence that we’re seeking. Figure fig. 1 gives an example of moving a Gaussian to another and shows three possible transportation maps.

While computing the Wasserstein metric using linear programming is in theory straightforward, the solution has quadratic size in the dimension of $X$, and state of the art algorithms [5, 46] can at best find a solution in $O(d^2)$ where $d$ is the dimension, time using advanced algorithms that exploit structure in the problem. Since $d$ is the number of pixels if used as a metric between images (interpreted as densities) or much higher if the metric is used between probability distributions of images this quickly becomes infeasible even at moderate image sizes. A wealth of methods have hence been developed to make computing the distance more feasible. One of the most successful such methods is to use Sinkhorn iterations [17, 37] which allows one to solve a version of eq. (3.0.1) where one adds a entropy term which ensures that the optimal $\pi^*$ has a low-dimensional representation. This allows us to find a approximation to the Wasserstein distance in $O(d^2)$ time. In the article Learning to solve inverse problems using Wasserstein loss we used this technique combined with another fast Fourier transform based trick that exploits translation invariance of the underlying metric to compute the Wasserstein distance in $O(d \log d)$ time. We also implemented this in a neural network and used it as a loss function for training a supervised solver to a inverse problem.

Another class of methods for computing the Wasserstein distance is to make use of the Kantorovich duality [84] which gives the following dual formulation:

$$D_{\text{Wasserstein}}(\mu_1, \mu_2) = \sup_D \mathbb{E}_{x_1 \sim \mu_1} D(x_1) - \mathbb{E}_{x_2 \sim \mu_2} D(x_2)$$

where the supremum is taken over all 1–Lipschitz functions $X \to \mathbb{R}$. This formulation lends itself particularly well for use with neural networks since the optimization over functions can be approximated with optimization over some class of neural networks [8]. The hard part is then to to enforce the 1–Lipschitz condition and the first article simply did this by limiting the weights of the network, a crude approximation.

A more exact method is to use the characterization of 1–Lipschitz functions in terms of their derivatives [22]. If $X$ is a Banach space, e.g. if $d(x_1, x_2) = \|x_1 - x_2\|$, it can be proven that $D$ is 1-Lipschitz if and only if $\|\partial D\|_* \leq 1$ where $\|\cdot\|_*$ is the dual norm. This was traditionally only done in the euclidean norm, but in the work Banach Wasserstein GAN we gave the full extension of the method to Banach spaces and applied it to GAN, where we learned a generative model for images.

The final paper of this thesis Deep Bayesian Inversion completes the circle by using these GAN to solve Bayesian inverse problems. In particular we learn how to generate samples from the posterior distribution by minimizing the Wasserstein
distance between the empirical distribution and the distribution generated by a neural network, where the Wasserstein distance was approximated using the Kantorovich duality.
References


3. Wasserstein Distances


3. **Wasserstein Distances**


3. **Wasserstein Distances**


[88] Christopher A Wong et al. “Deep neural networks learning to solve nonlinear inverse problems for the wave equation”. In: ().


Part II: Research Papers
Contributions

All of the papers in this thesis were the result of collaborations, some more extensive than others. My contributions were as follows.

Solving ill-posed inverse problems using iterative deep neural networks

The research idea was drafted together with Ozan Öktem and we developed the "learned iterative reconstruction" idea together. I proposed the practical architecture and did all implementation and evaluation. The paper was co-written with Ozan.

Learned Primal-Dual Reconstruction

The research idea came up in discussions with Ozan during the writing of the first paper. I then implemented the algorithm and wrote most of the paper with feedback from Ozan.

Task adapted reconstruction for inverse problems

I proposed the research idea. I also did some of the implementation, specifically with respect to the segmentation experiments. I developed the framing of the problem as a statistical decision theory problem with Ozan and the theoretical considerations with Olivier. The paper was co-written by all of the authors.

Learning to solve inverse problems using Wasserstein loss

I proposed the idea and most of the implementation and evaluation, following previous implementations by the other authors. The theoretical results were developed together with Axel Ringh and Johan Karlsson. The paper was co-written by all authors.

Banach Wasserstein GAN

I proposed the idea. The implementation, evaluation and theoretical results was done jointly with Sebastian Lunz. Both authors contributed equally to writing the paper.

Deep Bayesian Inversion

I proposed both the idea of using cGAN to sample from the posterior and the direct estimation method. I also did all implementation and evaluation. The theoretical results were done jointly with Ozan. Both authors contributed equally to writing the paper.
Solving ill-posed inverse problems using iterative deep neural networks

Jonas Adler¹,² and Ozan Öktem¹

¹ Department of Mathematics, KTH—Royal Institute of Technology, 100 44 Stockholm, Sweden
² Elekta AB, Box 7593, SE-103 93 Stockholm, Sweden

E-mail: ozan@kth.se

Received 22 May 2017, revised 22 September 2017
Accepted for publication 24 October 2017
Published 22 November 2017

Abstract
We propose a partially learned approach for the solution of ill-posed inverse problems with not necessarily linear forward operators. The method builds on ideas from classical regularisation theory and recent advances in deep learning to perform learning while making use of prior information about the inverse problem encoded in the forward operator, noise model and a regularising functional. The method results in a gradient-like iterative scheme, where the ‘gradient’ component is learned using a convolutional network that includes the gradients of the data discrepancy and regulariser as input in each iteration.

We present results of such a partially learned gradient scheme on a non-linear tomographic inversion problem with simulated data from both the Sheep-Logan phantom as well as a head CT. The outcome is compared against filtered backprojection and total variation reconstruction and the proposed method provides a 5.4 dB PSNR improvement over the total variation reconstruction while being significantly faster, giving reconstructions of 512 × 512 pixel images in about 0.4 s using a single graphics processing unit (GPU).

Keywords: tomography, deep learning, gradient descent, regularization

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

1. Introduction

Inverse problems refer to problems where one seeks to reconstruct parameters characterising the system under investigation from indirect observations. Such problems arise in several areas of science and engineering. Mathematically, an inverse problem can be formulated as reconstructing (estimating) a signal \( f_{true} \in X \) from data \( g \in Y \) where
\[ g = T(f_{\text{true}}) + \delta g. \]  

(1)

In the above, \( X \) and \( Y \) are topological vector spaces, \( T: X \to Y \) (forward operator) models how a given signal gives rise to data in absence of noise, and \( \delta g \in Y \) is a single sample of a \( Y \)-valued random variable that represents the noise component of data.

Many inverse problems, such as those arising in imaging, are naturally formulated when both signal and data are functions. In such case, \( X \) is some Banach/Hilbert space of functions defined on a fixed image domain \( \Omega \subset \mathbb{R}^d \) and \( Y \) is likewise a Banach/Hilbert space of functions defined on a fixed data manifold \( M \). An important remark here relates to the nature of the data manifold. It can be a subset of Euclidean space, but this is not necessarily the case. In fact, x-ray tomographic imaging leads to inverse problems where elements in the data manifold \( M \) represent lines in \( \mathbb{R}^d \).

1.1. Classical regularisation

A common approach in solving an inverse problem of the form in (1) is to minimise the miss-fit against data. For example by minimising

\[ f \to L(T(f), g), \]  

(2)

where \( L: Y \times Y \to \mathbb{R} \) is a suitable affine transformation of the data log-likelihood [6]. Then, one may interpret minimising the above as finding a maximum likelihood solution to (1).

This minimisation is a large scale optimisation problem that for typical choices of \( T \) is ill-posed, that is, a solution (if it exists) is unstable with respect to the data \( g \) in the sense that small changes to data results in large changes to a reconstruction. Hence, finding a maximum likelihood solution (there may be several) typically leads to over-fitting against data.

Within classical regularisation theory, there are currently three strategies for avoiding over-fitting when solving (1). One is approximate inverse that is applicable to cases when \( X \) has a mollifier. The idea is to construct a pseudo-inverse to \( T \) using the mollifier [36]. Another approach is iterative regularisation, which starts out by considering a fixed point iteration scheme for minimising (2). Over-fitting is avoided by stopping the iterates early, which is a feasible strategy if the iterates are semi-convergent [6, 16, 21, 23]. Finally, we have variational regularisation where over-fitting is avoided by introducing a functional \( S: X \to \mathbb{R} \) (regularisation functional) that encodes \textit{a priori} information about \( f_{\text{true}} \) and penalises unfeasible solutions [16, 35]. Hence, instead of minimising only the data discrepancy functional, one now seeks to minimise the regularised objective functional by solving

\[ \min_{f \in X} [L(T(f), g) + \lambda S(f)] \quad \text{for a fixed } \lambda \geq 0. \]  

(3)

In the above, \( \lambda \) (regularisation parameter) governs the influence of the \textit{a priori} knowledge encoded by the regularisation functional against the need to fit data.

Formulation in (3) suggests a ‘plug-and-play’ structure where the forward operator and data discrepancy functional can be adapted to the specific application, and the regularisation functional is chosen to capture \textit{a priori} information. This flexibility makes variational methods a powerful framework for image reconstruction, and especially so in cases when the main concern is reconstruction ‘quality’. A typical example in imaging is total variation (TV) regularisation, which is suitable for signals represented by scalar functions of bounded variation with sparse gradient. The corresponding regularisation functional is then given as \( S(f) := \| \nabla f \|_1 \).

There are however some drawbacks that come with using variational methods. One is that they are inherently computationally demanding, which is an issue in many applications.
Another is that the prior information needs to be encoded as an explicit functional, which limits the type of a priori information that can be accounted for. Finally, how to appropriately choose the regularisation parameter(s) is non-obvious for many applications.

1.2. Machine learning approaches to inverse problems

Machine learning is commonly used for non-linear function approximation under weak assumptions. Applied to the inverse problem in (1), it can be phrased as the problem of finding a (non-linear) mapping $T^{\dagger}_{\Theta}: Y \to X$ satisfying the following pseudo-inverse property:

$$T^{\dagger}_{\Theta}(g) \approx f_{\text{true}} \text{ whenever data } g \text{ is related to } f_{\text{true}} \text{ as in (1).}$$

A key element is to parametrise the set of such pseudo-inverse operators by $\Theta \in Z$, where $Z$ is a suitable parameter space. The ‘learning’ part refers to choosing an ‘optimal’ $\Theta$ given training data, where the concept of optimality is typically quantified through a loss functional that measures the ‘quality’ of a learned pseudo-inverse $T^{\dagger}_{\Theta}$.

The manner in which the loss functional is specified depends on the type of training data, and here we separate between supervised and unsupervised learning. These two approaches are fundamentally different and this article focuses on the supervised learning case since it is the problem with the most structure and we expect learning to give larger improvements over traditional methods.

1.2.1. Supervised learning. In supervised learning, training data are independent identically distributed realisations of a $(Y \times X)$-valued random variable $(g, f)$ with a known probability distribution $\chi$. Estimating $\Theta \in Z$ from training data can be formulated as minimising a loss functional $L(\Theta)$ that frequently has the following structure:

$$L(\Theta) := E_{\mu}[d(T^{\dagger}_{\Theta}(g), f)]. \quad (4)$$

In the above, $T^{\dagger}_{\Theta}: Y \to X$ is the pseudo-inverse that is given by $\Theta \in Z$, $d: X \times X \to R$ is a ‘distance’ function quantifying the quality of a specific reconstruction, and $F_{\mu}$ maps real-valued random variables on $Y \times X$ to real numbers.

A common choice is to use the expected loss w.r.t. the squared distance:

$$L(\Theta) := E_{\mu}[\|T^{\dagger}_{\Theta}(g) - f\|_{X}^{2}]. \quad (5)$$

One may also consider other loss functionals and the method we suggest can easily be adapted to these. As an example, a very conservative reconstruction method would use a loss functional given by the supremum of the $\infty$-norm:

$$L(\Theta) := \text{ess sup}_{\nu}[\|T^{\dagger}_{\Theta}(g) - f\|_{\infty}].$$

In practical applications, one needs to choose a distribution $\mu$ whose samples are representative for the application. As an example, if the application in question is computed tomography (CT) imaging of human heads, then samples from $\mu$ should be interpretable as human heads with associated noisy CT data.

Choosing $\mu$ such that the aforementioned requirement hold is naturally realised in three different ways. The first is to analytically specify $\mu$, this is, e.g. the case for the randomly generated ellipses in section 3. The second is to choose $\mu$ as the empirical distribution derived from available measurements. Finally, these two approaches can be mixed, e.g. by expressing

J Adler and O Öktem

Inverse Problems 33 (2017) 124007
\(\chi\) as an analytically known conditional density of \(g\) given \(f\) multiplied with the empirical density of \(f\). This is the case for the head CT reconstructions in section 3 where we use a finite number of heads and simulate CT data according to a physical noise model.

1.2.2. Unsupervised learning. In unsupervised learning there is no access to input-output pairs as in the supervised learning setting. Instead, here training data are given as elements in \(Y\). The corresponding mathematical setting is to consider these as independent identically distributed realisations of a \(Y\)-valued random variable \(g\) with a known probability density \(\mu\). A natural choice for a loss function is now to quantify how well the learned reconstruction matches the regularised data discrepancy, i.e.

\[
L(\Theta) := \mathbb{E}_\mu \left[ L\left( T\left( T_\Theta^\dagger(g) \right), g \right) + S\left( T_\Theta^\dagger(g) \right) \right].
\]

This can be interpreted as learning an optimiser for the variational regularisation in (3).

1.3. Survey of the field

Data driven approaches, and in particular deep learning using convolutional neural networks, have shown dramatic improvements over the state-of-the-art in several applications [26]. Likewise, knowledge-driven approaches outlined in section 1.1, and in particular variational regularisation, have provided a flexible toolbox for solving inverse problems.

As already mentioned, usage of variational regularisation is however associated with several challenges: computational feasibility, flexibility in the prior information that can be accounted for, and choice of regularisation parameter(s). A natural idea is thus to combine elements of deep learning and variational regularisation in order to address these challenges. This has been attempted by several authors as we now outline.

1.3.1. Fully learned reconstruction. Approaching the inverse problem directly with machine learning amounts to learning \(T_\Theta^\dagger: Y \rightarrow X\) from data such that it approximates an inverse of \(T\) in (1). An example of such an approach for solving very small scale tomographic reconstruction problems is given in [4, 31].

An obvious disadvantage with such fully learned approaches is that the result is likely to depend on the data manifold, so training data needs to be rich enough to account for all various data manifolds that one is likely to encounter. Furthermore, training data also needs to be rich enough to allow the learning scheme to learn the structure in \(T\), which is given by the physics laws governing the formation of data from a signal. Finally, in many applications the adequate digitalisations of the signal and data often requires very high dimensional arrays.

The above considerations imply that the parameter space \(Z\) used for parametrising possible inverse operators has to be very high dimensional in a fully learned approach. Therefore, the idea of learning \(T_\Theta^\dagger\) from data without using any knowledge of the physics or the data manifold quickly becomes in-feasible due to lack of training data.

1.3.2. Sequential data and knowledge driven reconstruction. The idea here is to separate the learned components from a part that encodes some knowledge about the structure of \(T\) and the data manifold. Formalising this, we assume

\[
T_\Theta^\dagger = B_\Theta \circ A \circ C_\Theta
\]
where $A: Y \to X$ is a known component that encodes knowledge about the structure of the forward operator $T$ whereas the operators $B_\Theta: X \to X$ and $C_\Theta: Y \to Y$ are the learned components.

An important special case is when $C_\Theta$ is not learnt, which entirely separates the computation of $g \mapsto A(g)$ from the learning of $B_\Theta$, i.e. the original inverse problem in (1) is recast as learning $B_\Theta: X \to X$. This significantly simplifies the implementation since the often demanding task of computing $g \mapsto A(g)$ can be separated from the learning software. It also ensures the data manifold is not explicitly part of the learning. One may furthermore exploit additional structure, like locality and/or invariance of the operator $A \circ T$. A key step for such a sequential data and knowledge driven reconstruction scheme is to have candidates for $A$ and one natural option is to let it be some pseudo-inverse. As an example, in tomographic applications it can be given by the back-projection or the filtered back-projection operator. Next, when learning $B_\Theta$ from data, one may use approaches that build on the corpus of knowledge that exists for denoising signals in $X$. An example demonstrating this approach for tomographic reconstruction is given in [20, 22].

On the other hand, for ill-posed inverse problems some information is irreversibly lost when making the assumption in (6) since the learned operators $C_\Theta$ and $B_\Theta$ cannot recover information that is lost by using a pseudo-inverse $A$. To alleviate this problem, for linear forward operators we can consider choosing $A$ as the adjoint of the forward operator, i.e. $A := T^* : Y \to X$. This can be seen as learning to solve the normal equations for (1) since

$$f_{\text{true}} \approx B_\Theta(T^*(g)) \implies f_{\text{true}} \approx B_\Theta(T^*(T(f_{\text{true}}))) \implies B_\Theta \approx (T^* \circ T)^{-1}.$$ 

Nonetheless, solving the normal equations for ill-posed problems is often more ill-posed than the original inverse problem, so such a learning procedure would need to include some kind of regularisation.

We conclude by pointing to examples where the operator $C_\Theta : Y \to Y$ is learned. One such case is [41] where tomographic reconstruction is performed by learning $C_\Theta$. Another similar, but more advanced, example is [32] where the operator $A$ is given by several filtered backprojection (FBP) operators and the learned operator $C_\Theta$ is given by the filter coefficients.

While such approaches can in principle address all of the aforementioned issues with variational methods, they are ultimately limited by what $A: Y \to X$ manages to capture about the ‘inverse’ of $T: X \to Y$. In conclusion the final reconstruction cannot contain information that is not already present in $A(g)$.

### 1.3.3. Learning for variational reconstruction.

In these methods the aim is to solve a variational problem by using techniques from machine learning.

One option is to use the latter in order to select the regularisation parameter(s). This procedure can be re-formulated as a bi-level optimisation scheme whose mathematical properties (like existence) can be analysed in a functional analytic setting [8, 9, 12–14, 25]. On the other hand any implementation of an iterative scheme for solving the aforementioned parametrised optimisation problem will terminate after a finite number of iterates. Hence, the outcome will not only depend on the formulation of the parametrised optimisation problem, but also on the solution scheme one chooses to use. In conclusion, a bi-level optimisation scheme of the above type does not by itself uniquely determine a reconstruction operator.

Another approach replaces the learning of the optimisation problem with learning an optimiser adapted to a given class of optimisation problems. In [3] a stochastic gradient descent method is learned from training data consisting of optimisation problems, each associated with a deep learning problem. The output is thus a trained stochastic gradient descent
method that can be used to train other deep neural networks. This overall 'learning to learn' approach can in principle be extended to other use cases, such as solving inverse problems, by merely changing the underlying class of training data. While this approach may to some extent address the performance issues associated with variational methods, it is in the same way limited in the types of priors that can be represented.

1.3.4. Learned iterative reconstruction. The idea here is to refrain from formulating an optimisation problem without compromising upon the ability to account for knowledge about the inverse problem, such as forward operator, data discrepancy, regulariser, etc.

An example of such a scheme for solving inverse problems is [42], which learns an ADMM-like scheme for Fourier inversion. Another is [11], which learns a ‘proximal’ in an ADMM-like scheme for various image processing problems. Finally, [33] considers solving finite dimensional linear inverse problems typically arising in image restoration. The idea is to learn over a broader class of schemes instead of restrict attention to a specific type of scheme, like ADMM above.

Similar to the sequential data and knowledge driven reconstruction methods, the learned iterative reconstruction schemes can address issues related computational feasibility, flexibility in the prior information that can be accounted for, and choice of regularisation parameter(s). On the other hand, they utilise the same \( a \text{ priori} \) information as in variational regularisation.

1.4. Contribution and overview of paper

This paper develops a framework for learned iterative reconstruction, generalising the ideas in [33] along several directions. First, we consider solving inverse problems with (possibly) non-linear forward operators. Next, our framework is formulated in a coordinate free functional analytic setting. Furthermore, we consider the issue of proper initialisation and allow for further prior knowledge to be included in the form of a regulariser. Finally, we provide a generic and scalable open source implementation of our method based on operator discretization library (ODL) [2] that can be applied to a wide range of realistic inverse problems. This also includes the trained parameter \( \Theta \) used for generating the results shown in this paper. To show that the approach can handle (non-linear) forward operators in large scale inverse problems, we consider tomographic reconstruction with a non-linear ray transform inversion.

Section 2 derives a partially learned gradient decent scheme for solving (1) in the functional analytic setting. This section also introduces the deep convolutional network that is used later for tomographic reconstruction. Section 3.1 describes the implementation of the partially learned gradient decent scheme in section 2 and software components used for computations. Section 3 tests the performance of the partially learned gradient decent scheme on tomographic inverse problems. The paper concludes with a discussion in section 4 and a summary of future work and conclusions is given in section 5.

2. Solving inverse problems by learned gradient descent

We begin by proving a heuristic motivation that comes from comparing two natural considerations involving gradient descent schemes associated with solving (1). This results in an initial scheme for partially learned gradient descent given in algorithm 1, which is then extended by adding persistent memory and resulting in the scheme in algorithm 2. Next is a description of

how to integrate deep learning with algorithm 2, resulting in the final scheme given in algorithm 3.

2.1. Motivation

The starting point in learning an iterative scheme that combines elements from (deep) machine learning and classical regularisation theory is to consider the error functional $E: X \to \mathbb{R}$ defined as

$$E(f) := d(f, f_{true})$$

where $d: X \times X \to \mathbb{R}$ is the distance functional that appears in the definition of the loss functional (4) used for training. It measures how good well $f$ approximates $f_{true}$, and one natural error functional corresponding to (5) is

$$E(f) = \|f - f_{true}\|^2_X.$$

Ideally, solving (1) would be based on minimising the error functional, which for obvious reasons is not possible. Hence, we need to use a substitute. In variational regularisation theory, such a substitute is given by the regularised objective functional in (3). Much of regularisation theory aims at choosing the objective functional in (3) so that the regularised solution approximates the true signal to be recovered:

$$\arg\min_{f \in X} E(f) \approx \arg\min_{f \in X} \left[ \mathcal{L}(T(f), g) + \lambda \mathcal{S}(f) \right]. \tag{7}$$

Assume next that the objective functional in the right hand side of (7) is Fréchet differentiable and (strictly) convex. Then, a simple gradient descent scheme could be used to find a minimum:

$$f_i := f_{i-1} - \sigma \left( \nabla \mathcal{L}(T(\cdot), g)[f_{i-1}] + \lambda \nabla \mathcal{S}(f_{i-1}) \right) \tag{8}$$

where, assuming a differentiable likelihood and forward operator, we note that

$$\nabla \mathcal{L}(T(\cdot), g)[f] = \left[ \partial T(f) \right]^* \left( \nabla \mathcal{L}(\cdot, g)(T(f)) \right) \quad \text{for any } f \in X.$$

Likewise, considering the left hand side in the same way, a differentiable convex error functional would allow one to use a corresponding scheme for finding a minimum:

$$f_{i+1} := f_i - \sigma \nabla E(f_i). \tag{9}$$

Since the gradient mapping $\nabla E: X \to X$ in (9) requires knowledge about the true signal, it is natural to try to learn it from training data while utilising knowledge about the gradient mappings $\nabla \mathcal{L}(T(\cdot), g), \nabla \mathcal{S}: X \to X$. For this purpose, we introduce the (learned) updating operator $\Lambda_{\Theta}: X \times X \times X \to X$ that, given an appropriately selected (learned) parameter $\Theta \in Z$, should satisfy

$$\Lambda_{\Theta} \left( f, \nabla \mathcal{L}(T(\cdot), g)[f], \lambda \nabla \mathcal{S}(f) \right) \approx \nabla E(f).$$

These considerations suggests a partially learned gradient descent scheme specified as in algorithm 1.
Algorithm 1. Initial partially learned gradient descent.

1: Select an initial guess $f_0$
2: for $i = 1, \ldots, I$ do
3: $\Delta f_i \leftarrow -\sigma \Lambda \Theta \{ f_{i-1}, \nabla [\mathcal{L}(\cdot, g)](f_{i-1}), \nabla S(f_{i-1}) \}$
4: $f_i \leftarrow f_{i-1} + \Delta f_i$
5: $T^\dagger \Theta (g) \leftarrow f_I$

Algorithm 1 suffers from several unnecessary shortcomings that are easily addressed. The regularisation parameter $\lambda$ and the step length $\sigma$ have to be explicitly chosen, a task that is known to be troublesome in practical applications (section 4.2). One may instead make these part of $\Theta$ and thereby learn them from training data. Next, the convergence rate of gradient descent schemes can be accelerated by using information from previous iterates (memory) as in quasi-Newton schemes [27]. For this purpose we introduce persistent memory $s \in X^M$ that allows algorithm 1 to use information from earlier iterates. The learned updating operator now becomes a mapping

$$ \Lambda_{\Theta} : X^M \times X \times X \times X \rightarrow X^M \times X. $$

Finally, one often also has the possibility to select the initial iterate $f_0$ using some suitable pseudo-inverse $T^\dagger : Y \rightarrow X$. Considering these modifications results in the partially learned gradient descent scheme listed in algorithm 2.

Algorithm 2. Partially learned gradient descent.

1: $f_0 \leftarrow T^\dagger (g)$
2: Initialize ‘memory’ $s_0 \in X^M$
3: for $i = 1, \ldots, I$ do
4: $(s_i, \Delta f_i) \leftarrow \Lambda \Theta \{ s_{i-1}, f_{i-1}, \nabla [\mathcal{L}(\cdot, g)](f_{i-1}), \nabla S(f_{i-1}) \}$
5: $f_i \leftarrow f_{i-1} + \Delta f_i$
6: $T^\dagger \Theta (g) \leftarrow f_I$

2.2. Parametrising the learned updating operators

The goal here is to specify the class of learned updating operators that are parametrised by $\Theta \in Z$. Following the paradigm in (deep) neural networks, we start by defining a family of affine operators

$$ \mathcal{W}_{w_n, b_n} : X^{c_{n-1}} \rightarrow X^{c_n} \quad \text{for} \quad n = 0, \ldots, N, $$

(11)

parametrised by linear mappings $w_n : X^{c_{n-1}} \rightarrow X^{c_n}$ (weights) and $b_n \in X^{c_n}$ (biases). Here, $N$ is usually referred to as the depth of the neural network that will eventually define the learned updating operator and $c_n$ is the number of channels in the $n$:th layer. Next, we introduce a family of non-linear operators

$$ \mathcal{A}_n : X^{c_n} \rightarrow X^{c_n} $$

(12)

that are given by point-wise application of a fixed non-linear scalar function, henceforth called the response function.

By chaining compositions, we now define a parametrised family of learned updating operators as

$$ \Lambda_{\Theta} := (A_0 \circ \mathcal{W}_{w_N, b_N}) \circ \cdots \circ (A_1 \circ \mathcal{W}_{w_1, b_1}) $$
with \( \Theta := \{(w_N, b_N), \ldots, (w_1, b_1)\} \). In order to match the domain and range of the operator in (10), we need to assume that \( c_0 = M + 3 \) and \( c_N = M + 1 \).

Such parametrised operators are used in machine learning applications for two primary reasons: computability and descriptive power. In order to learn the parameters \( \Theta \) from training data, a (stochastic) gradient descent method is typically applied in which case the derivative \( \frac{\partial \Lambda_\Theta}{\partial \Theta} \) and its adjoint \( \left[\frac{\partial \Lambda_\Theta}{\partial \Theta}\right]^* \) needs to be repeatedly computed and here one may use the chain rule. This becomes particularly easy to perform in a computationally feasible manner for learned updating operators of this form. Furthermore, introducing the non-linear component in (12) allows the learned operator to approximate a large set of non-linear operators [19].

2.2.1. Choice of affine and non-linear operator families. Our next step is to further narrow down the generative models for the operator families \( \mathcal{W}_{\nu, b} \) and \( \mathcal{A} \). We start by writing the affine operator in (11) as

\[
\mathcal{W}_{\nu, b} = (\mathcal{W}_{\nu, b}^1, \ldots, \mathcal{W}_{\nu, b}^{c_N})
\]

where the components

\[
\mathcal{W}_{\nu, b}^l : X^{c_{n-1}} \rightarrow X \quad \text{for } l = 1, \ldots, c_n
\]

represent the affine transformation for the \( l \)th channel in the \( n \)th layer. In particular, using linearity these can be represented as

\[
\mathcal{W}_{\nu, b}^l (f_1, \ldots, f_{c_{n-1}}) = b_l + \sum_{j=1}^{c_{n-1}} w_{l,n}^j (f_j)
\]

where \( w_{l,n}^j : X \rightarrow X \) is a channel-wise linear operator.

However, optimizing over the set of all linear operators would result in a very large number of parameters. A solution is to assume that \( w_{l,n}^j \) is translation invariant, which indicates that we seek to use a convolutional network architecture.

Convolution operators are useful for representing translation invariant image features at a specific scale that is governed by the size of the support of the kernel. Hence, hierarchically organising such operators with small kernels in layers provides a (deep) convolutional neural network (CNN) with vastly reduced number of network parameters that is efficient to implement, and at the same time capable of representing various image features at different scales.

In the case of tomographic reconstruction, the situation is however different since there are many non-local dependencies (all pixels/voxels on a line contribute to the value of the ray-transform). Thus, applying convolution network architectures to directly learn the reconstruction would be problematic. On the other hand, the scheme outlined in algorithm 2 includes the forward operator that accounts for these global dependencies. For this reason, it is feasible to use convolution network architectures to learn the learned updating operator in (10).

By using convolution operators, we find that the affine operators can be written

\[
\mathcal{W}_{\nu, b}^l (f_1, \ldots, f_{c_{n-1}}) = b'_l + \sum_{j=1}^{c_{n-1}} w_{l,n}^j \ast f_j
\]

where \( b'_l \in \mathbb{R} \) represents the bias and \( w_{l,n}^j \) is given as a ‘matrix’ of convolution kernels \( w_{l,n}^j \in \mathbb{R} \). Hence, our parameter space becomes

\[
Z = (X^{c_0 \times c_{n-1}} \times \mathbb{R}^{c_0}) \times \ldots \times (X^{c_1 \times c_0} \times \mathbb{R}^{c_1})
\]
Finally, the non-linear response functions $A_i$ in (12) can be chosen in different ways and we will be using the rectified linear unit (ReLU) [29]

$$ \text{relu}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{else} \end{cases}. $$

2.3. The partially learned gradient descent algorithm

A number of hyper-parameters needs to be chosen prior to learning. These are the number of layers $N \in \mathbb{N}$, the number of channels $c_1, \ldots, c_{N-1} \in \mathbb{N}$ in each layer, the number of iterations $I$, and the size of the memory $M$.

In the examples shown in in section 3, we let the weights $w_i$ be represented by $3 \times 3$ pixel convolutions and we used $N = 3$ layers. The number of convolutions in each layer was selected as $m_1 = 32$ and $m_2 = 32$. We selected the number of iterations to be $I = 10$ and the amount of memory to be $M = 5$. Such low numbers were selected in order to reduce the space of allowed parameters which in turn should help reduce over-fitting. All parameters were selected by simple trial and error and it is likely that a more sophisticated set-up would give better results or be better suited for a particular application. For further details, see the supplemental source code.

Once the hyper-parameters are chosen, one may now fully specify the partially learned gradient descent, which is done in algorithm 3. The scheme learns the updating operator by learning the scalars $b_l \in \mathbb{R}$ and the convolution kernels (functions) $w_j^l$ from training data. The resulting learned updating operator can then be used to solve the inverse problem in (1).

Algorithm 3. Partially learned gradient descent.

1: $x_0 \leftarrow 0$
2: $f_0 \leftarrow T^\dagger(g)$
3: for $i = 1, \ldots, I$ do
4: $u_1^i \leftarrow \{f_{i-1}, x_{i-1}, \nabla L(T(\cdot,g))(f_{i-1}), \nabla S(f_{i-1})\}$
5: $u_2^i \leftarrow \text{relu}(W_{w,1}(u_1^i))$
6: $u_3^i \leftarrow \text{relu}(W_{w,2}(u_2^i))$
7: $(a_4^i, \Delta L) \leftarrow W_{w,3}(u_3^i)$
8: $s_i \leftarrow \text{relu}(a_4^i)$
9: $f_i \leftarrow f_{i-1} + \Delta f_i$
10: $T^*_d(g) \leftarrow f_I$

3. Implementation and evaluation

The algorithm was tested on the two-dimensional computed tomography problem. The signal is in this case real valued functions defined on a domain in $\mathbb{R}^2$ representing images and $X$ is a suitable vector space of such functions. The corresponding forward operator is expressible in terms of the ray transform $P : X \to Y$, which integrates the signal over a set of lines $M$ given by the acquisition geometry. Hence, elements in $Y$ are functions on lines

$$ P(f)(\ell) = \int f(x)dx \quad \text{for } \ell \in M. $$
As training data we consider CT simulations from two particular types of phantoms with different forward operators and noise models:

**Ellipses:** Training data is randomly generated ellipses on a 128 × 128 pixel domain. The projection geometry was selected as a sparse 30 view parallel beam geometry with 5% additive Gaussian noise added to the projections. In this case, the log-likelihood was selected as the squared $L^2$ norm $L(\cdot, g) := \frac{1}{2} \| \cdot - g \|_2^2$ which implies

$$\nabla [L(P(\cdot), g)](f) = P^* (P(f) - g)$$

The phantoms were generated 'on the fly', giving an effectively infinite dataset.

**Heads:** The training is simulated projections of 512 × 512 pixel, 256 × 256 mm slices of CT scans of human heads as provided by Elekta (Elekta AB, Stockholm, Sweden). The acquisition geometry defining the data manifold was selected as a fan beam geometry with source-axis distance of 500 mm, source-detector distance 1000 mm, 1000 pixel, and 1000 angles.

Here, in order to get a accurate noise model we used a non-linear forward operator given by

$$T(f)(\lambda) = \lambda \exp\left(-\mu P(f)(\lambda)\right)$$

where $\lambda \in \mathbb{R}^+$ is the mean number of photons per pixel, taken to be 10 000, and $\mu \in \mathbb{R}^+$ is the linear attenuation coefficient which was taken to be that of water ($\approx 0.2$ cm$^{-1}$). Poisson noise was added to the projections, and given 10 000 photons per pixel, which corresponds to a low dose scan. For this type of noise, the log-likelihood is given by the Kullback–Leibler divergence and the data discrepancy becomes

$$L(T(\cdot), g) := \int_{\mathbb{R}} \left[ T(f)(\lambda) + g(\lambda) \log \left( \frac{g(\lambda)}{T(f)(\lambda)} \right) \right] d\lambda$$

which implies that

$$\nabla [L(T(\cdot), g)](f) = \left[ \partial T(f) \right]^* \left( 1.0 - \frac{g}{T(f)} \right).$$

In the above, the adjoint of the derivative of the forward operator applied in a perturbation $\delta g \in Y$ is given by

$$\left[ \partial T(f) \right]^* (\delta g) = P^* \left( -\mu \lambda \exp\left(-\mu P(f)(\cdot)\right) \frac{\delta g(\cdot)}{T(f(\cdot))} \right).$$

which after some simplifications gives the following expression for the gradient:

$$\nabla [L(T(\cdot), g)](f) = -\mu P^* (T(f) - g) \quad \text{for } f \in X.$$

The training used 500 CT scans with a total of 41 000 slices.

For both cases the regulariser was selected as the Dirichlet energy, e.g.

$$S(f) := \frac{1}{2} \| \nabla f \|_2^2 \implies \nabla S(f) = \nabla \left( \nabla f \right)$$
which acts as a smoothing, thereby reducing noise [7]. See figure 1 for examples of the data used for training and validation.

3.1. Implementation

The methods described above were implemented in Python using ODL [2] and Tensorflow [1]. All operator-related components, such as the forward operator $T$, were implemented in ODL, and these were then converted into Tensorflow layers using the `as_tensorflow_layer` functionality of ODL. The neural network layers and training were implemented using Tensorflow.

The implementation utilises abstract ODL structures for representing functional analytic notions and is therefore generic, yet easily adaptable to other inverse problems. We used the ODL operator `RayTransform` in order to evaluate the ray transform and its adjoint using the GPU accelerated ‘astra_gpu’ backend [40]. The pseudo-inverse $T^\dagger$ was given by the filtered back-projection algorithm implemented in ODL as `fbp_op` with no additional smoothing filter.

We emphasise that the functional analytic formulation of algorithm 3 is critical to handle problems of this scale. As an example, storing the ray transform used for the heads dataset as a sparse matrix of floating point numbers would require about 1 GB of GPU memory.

3.1.1. Training. We trained the parameters $\Theta$ using the RMSPropOptimizer optimiser in Tensorflow. We initially used $10^5$ batches on the ellipses problem, where each batch contained 20 tomography problems with a learning rate starting at $10^{-3}$ and decayed according to the inverse of the iteration number down to about $10^{-5}$. This training took four days on a workstation with a single Nvidia GTX Titan GPU. These parameters were then used as an initial guess for the heads problem, once again trained according to the same scheme but with the learning rate starting at $10^{-5}$ and decreased to about $10^{-7}$ and with each batch containing only one tomographic problems due to memory limitations of the current implementation. This training took four days on the aforementioned hardware.

3.1.2. Comparison. We compare the performance of the partially learned iterative algorithm against the FBP algorithm and TV regularisation. The FBP reconstructions were performed using a Hann filter with bandwidth selected to maximise peak signal to noise ratio (PSNR). The data discrepancy in the TV regularisation matched the one used in the partially learned algorithm and the regularisation parameter was selected to maximise PSNR.

We solve the TV regularised problem without smoothing using the the generic ODL implementation of the non-linear primal dual hybrid gradient (PDHG) optimisation method [39]. This is needed since the forward operator in the heads dataset is non-linear. We used 1000 iterations at which point the objective function was stationary. For the ellipses dataset, the evaluation was performed on the modified Shepp–Logan phantom, while on the heads dataset a slice through the nasal region was used.

3.2. Results

We compare the reconstructions of the partially learned algorithm with the FBP and TV reconstructions for both the ellipse and head datasets and computed the PSNR, runtime and performed a visual comparison. The quantitative results are given in table 1, which visualisations are available in figures 2 and 3. We also display some partial results of the iterative algorithm for reference in figure 4.
We note that for the ellipse data, the FBP algorithm performs very poorly under the high noise while the TV and learned methods give comparable results. This is expected given that TV regularisation is very competitive for images of this type, nonetheless the learned method does outperform the TV method by approximately 2 dB and the visual result looks slightly more appealing, with less randomly occurring structures and significantly less stair-casing.

For the head dataset where the noise is lower, the filtered back-projection reconstruction performs much better, and is arguably comparable to the TV regularised reconstruction. The learned reconstruction provides (perhaps too) smooth images, where we note that especially in the boundary regions, e.g. in the air-skin boundaries and around the bone the algorithm performs amiably.

In addition to the visual results, we see that the learned algorithm is significantly better than the TV reconstruction w.r.t the PSNR, giving an >5 dB improvement. Finally, the runtime of
the algorithm, while being slightly slower than traditional filtered-back-projection, is significantly faster than the TV method as shown in Table 1.

3.3. Impact of including gradient mappings

The impact of including the gradient mappings

$$\nabla \mathcal{L}(T(\cdot), g) \mid \nabla S : X \rightarrow X$$
in the partially learned gradient scheme can be empirically analysed. We do this by training the network in the exact same manner with and without the gradients added and then performing 100 reconstructions of the Shepp–Logan phantom with the respective methods.

Without the gradients, the PSNR was 29.65 dB while it was 30.51 dB with the gradient of the data discrepancy. This can be compared to 32.02 dB with both gradients. Visual inspection also indicates that adding the gradients provides a sharper reconstruction with more detail, where especially in the case of no gradients the small inserts are barely visible. Performance wise, the method took 19 ms without the gradients, 64 ms with the gradient of the data discrepancy and 66 ms with both gradients. See figure 5 for a visual comparison.

Figure 3. Reconstructing a head phantom using FBP, TV and the partially learned gradient scheme. Data is simulated from a physiological head phantom, which includes some weak streaks (so these are part of the ground truth). All images are shown using a window set to [-200, 200] HU.
4. Discussion

The partially learned gradient scheme differs significantly from the current paradigm for regularisation of inverse problems, so there are several remarks that deserve a closer discussion.

4.1. Theory

The partially learned gradient scheme is presented with a strong emphasis on the algorithmic aspects, its implementation and its performance. There are however several interesting theoretical issues that deserve closer attention.
4.1.1. Deep neural networks in function spaces. The scheme in algorithm 2 is formulated in a functional analytic setting. However, the theory for deep neural networks is not well established in the infinite dimensional setting and there are several open issues that remain to be answered, such as determining what class of operators can be approximated by a given deep neural network and to what accuracy [19].

Another aspect relates to usage of probabilistic notions in infinite dimensional vector spaces. The classical theory for deep learning deals with finite, or at most countable data where the law of large numbers holds. In the infinite dimensional setting one needs to be more careful regarding which topologies that are used. It is clearly advantageous to work with spaces where one can prove various forms of weak convergence of probability measures and state and prove results corresponding to the law of large numbers, see [37, 38]. Hence, applying deep learning to infinite dimensional spaces is associated with a number of fundamental questions regarding convergence of the learning, and if it converges, in what sense?

4.1.2. Regularising properties. A theoretical topic of interest is to prove that the given reconstruction scheme constitutes a formal regularisation in the sense of [35], that is proving existence, stability and convergence.

Figure 5. Comparison of reconstructions using the partially learned gradient scheme with and without the gradient information. Note that gradient of data discrepancy includes the derivative of the forward operator.
Existence for $T^\dagger_{\Theta}: Y \to X$ is a non-issue since it this operator is given by a finite number of compositions of well-defined operators. Next, assume that the gradients of the data discrepancy $L\{T(\cdot), g\} : X \to \mathbb{R}$ and the regulariser $S: X \to \mathbb{R}$ are Lipschitz continuous. Then the partially learned pseudo-inverse $T^\dagger_{\Theta}$ is also Lipschitz continuous, which in turn implies stability. The final consideration concerns convergence, which is formally defined as

$$
\left\| T^\dagger_{\Theta} (T(f^{\text{true}}) + \delta g) - f^* \right\|_X \to 0 \quad \text{whenever} \quad \left\| \delta g \right\|_Y \to 0
$$

for some parameter choice rule for the hyper-parameters in section 2.2 and the training data, which uniquely define $\Theta$, and where $f^* \in X$ is some minimum norm solution to (1). Clearly, the above convergence criteria can only be satisfied in general if the hyper-parameters and training data used for learning $\Theta$ are re-chosen as the data noise tends to zero. To conclude, in figure 6 we compare the result of applying the partially learned reconstruction scheme to noiseless data while training the parameter $\Theta$ against noisy data. It is reasonable to expect a significantly lower error if the method was re-trained on noiseless data, but we are currently unable to give a rigorous proof that this would converge to zero.

4.2. Use cases

The framework for partially learned reconstruction was primarily motivated by a number of use cases involving ill-posed inverse problems. A number of challenges naturally arise when classical regularisation is applied to solve the associated inverse problems and below we describe how these challenges can be resolved using a partially learned reconstruction scheme.

4.2.1. Computational feasibility. The forward operator is an important part of a regularisation and the more accurately it models the relation between signal and data, the better the outcome. Usage of more accurate forward models is however almost always computationally more demanding. Likewise, more elaborate regularisation schemes that are better at utilising the available a priori knowledge are often also computationally more demeaning. As an example, several of the more advanced regularisers in the literature exploit some kind of sparsity using a $L_1$-like norm [35]. Such regularisers typically give rise to non-differentiable...
objective functional that require using optimisation algorithms from non-smooth analysis for their efficient solution. Finally, there may also be reconstruction parameters that, unlike regularisation parameter(s), do not influence the reconstruction quality. The role of the reconstruction parameters is to ensure the method is as efficient as possible, so these affect the speed of reconstruction.

Computational feasibility becomes especially critical in imaging applications since these involve very large-scale data structures. In such setting, variational and iterative regularisation schemes quickly become infeasible even for applications with moderate time requirements despite usage of state of the art algorithms.

The learned method algorithm 2 improves upon this by having an \textit{a priori} defined run-time which can be tweaked by using more or less iterates or a more complicated updating operator. By learning, we thus learn an optimal reconstruction scheme for a given execution time. Note that the run-time of the method on our examples is significantly faster than the TV regularised method.

4.2.2. Nuisance parameters. Nuisance parameters are additional unknowns that need to be reconstructed alongside the signal. They are not of primary interest, but they nevertheless need to be reconstructed. As an example, in certain tomographic applications the acquisition geometry (sampling of the data manifold $\mathcal{M}$) is partially unknown, so the nuisance parameters would be those needed for a precise description of said geometry. Another is use of a more accurate forward model, which often introduces nuisance parameters.

A common approach is to adopt an intertwined scheme in which each iterate involves updating the signal by reconstructing it from data using the previous value for the nuisance parameter(s), followed by updating the nuisance parameter(s) by reconstructing them from data and making use of the previous value of recently updated signal.

Our learned reconstruction scheme algorithm 2 can easily be extended to include such intertwined schemes.

4.2.3. Regularisation parameter selection rule. Regularisation parameter(s) govern the balancing between preventing over-fitting against the need to have a solution that generates data, which is consistent with measurements. To have an appropriate parameter choice rule is critical for success.

Unfortunately, there is little theory to guide how to choose the regularisation parameter(s). Mathematical results often study asymptotic behaviour of a parameter choice rule as data noise level tends to zero. Results mainly cover the case when noise in data is additive Gaussian and its magnitude can be reliably estimated [16], even though there are extensions for other noise types as well. Nevertheless, many of these assumptions are often not met in reality.

Some work has been done in selecting an optimal parameter using learning [8]. The proposed method encompasses this since the regularisation parameter (and other optimisation related parameters) are included in the learned updating operator and thus optimally selected from the training data.

4.2.4. Feature reconstruction. Reconstructing the signal is in many applications merely one part of a more elaborate scheme of transforming measured data to knowledge. As an example, in tomographic imaging the reconstructed image serves as input for an image analysis part. The latter often involves complex procedures, like segmentation and object recognition, that currently require involvement of human expertise.
There is a growing trend in including some of these into the inverse problem that is referred to as feature reconstruction. To some extent, compressed sensing can be seen as an example of feature reconstruction where the sparse coding dictionary is the feature extraction part. Other examples are joint image reconstruction and segmentation [5, 28, 34] and shape based reconstruction [17, 18, 30]. Such feature reconstruction methods are however hard to analyse theoretically and current methods are limited in the type of feature extraction capabilities they can include. They also tend to be computationally demanding.

It is natural to perform feature reconstruction by adding a feature extraction network to the learned reconstruction scheme such as in [15]. The proposed framework could in a similar way be extended to feature reconstruction by composing the learned reconstruction operator $T_\Theta$ with a feature extraction operator $R: X \to F$ where $F$ is a vector space of features. If the latter is differentiable, which is the case for deep learning based feature extractors, then we can define the loss functional (4) using the composed operator $R \circ T_\Theta$. This allows for truly end-to-end optimisation of task dependent reconstruction schemes for general inverse problems.

4.3. Stability

A general question often asked when learning is applied to some problem is whether the method generalises to other problems, e.g. if a method that is trained on a specific dataset can be applied to another dataset or to what extent one can change to forward operator without re-training.

Note first that the partially learned gradient scheme does not have an explicit regularisation parameter, instead its regularisation properties are implicitly contained in the training dataset (and to some extent in the hyper-parameters). Hence, a significant change in the training dataset (notably, a change of scaling) would require a re-training. On the other hand, empirical numerical experience suggests that dependence is relatively weak, at least for the tomographic reconstruction problems we considered. Specifically, we were able to successfully pre-train the system using a simplified acquisition geometry, a linearised forward operator, different domain size and significantly simplified phantoms and then successfully use this to train the network for the much more complicated heads dataset.

Finally, numerical experiments also suggests that changing the forward operator requires only a modest fine-tuning where the given parameters $\Theta$ can be used as an initial guess.

5. Conclusion and future work

We have presented a partially learned approach for solving ill-posed inverse problems that can integrate prior knowledge about the inverse problem with learning from training data. The presented method works with any non-linear operator and the method could easily be applied to a wide range of problems. Numerical experiments on tomographic data shows that the method gives notably better reconstructions than traditional FBP and TV regularisation. Furthermore, adding prior information improves the reconstruction. In conclusion, using prior knowledge about the forward operator, data acquisition, data noise model and regulariser can significantly improve the performance of deep learning based approaches for solving inverse problems, and especially so when the available training data is much smaller than the size of the parameter space.

An obvious next step is to tackle fully three-dimensional tomographic problems while training on two-dimensional datasets. It would also be of interest to improve upon the choice
of regulariser by adding more regularisers and/or more advanced regularisers such as wavelet based regularisers. Other more elaborate extensions are outlined below.

5.1. Extension to other iterative schemes

The given iterative method is based of the gradient descent scheme, but this scheme is known to be sub-optimal in the case of non-differentiable objective functions. A natural extension of the scheme is thus to instead consider iterative schemes better suited for this use case. One such iterative scheme is the (non-linear) PDHG algorithm [10, 39] for solving problems of the form

\[
\min_{f \in X} \left[ \mathcal{F}(K(f)) + \mathcal{G}(f) \right]
\]

where \( K: X \to U \) is a (possibly non-linear) operator between Banach spaces \( X \) and \( U \). The scheme is given by algorithm 4 and the proximal operators in algorithm 4 are given by

\[
\text{prox}_{\sigma \mathcal{F}^*}(h) = \arg \min_{h' \in U} \left[ \mathcal{F}^*(h') + \frac{1}{2\sigma} \| h' - h \|^2_U \right]
\]

\[
\text{prox}_{\tau \mathcal{G}}(f) = \arg \min_{f' \in X} \left[ \mathcal{G}(f') + \frac{1}{2\tau} \| f' - f \|^2_X \right]
\]

where \( \mathcal{F}^* \) is the Fenchel conjugate of \( \mathcal{F} \). The special case of TV regularised reconstruction for (1) amounts to selecting \( K: X \to Y \times X^d \) as

\[
K(f) := \left[ \langle f \rangle, \| f \|_2, 0 \right].
\]

The resulting algorithm is summarised in algorithm 4.

**Algorithm 4.** Non-linear PDHG algorithm.

1: Given: \( \sigma, \tau > 0 \) s.t. \( \sigma \tau \| K \|^2 < 1 \), \( \theta \in [0, 1] \) and \( f_0, h_0 \in U \).
2: for \( i = 1, \ldots, J \) do
3: \( h^{i+1} = \text{prox}_{\sigma \mathcal{F}^*}(h^i + \theta K(f^i)) \)
4: \( f^{i+1} = \text{prox}_{\tau \mathcal{G}}(f^i - \tau [\partial \mathcal{G}(f^i)'(h^{i+1})]) \)
5: \( f^{i+1} = f^{i+1} + \theta (f^{i+1} - f^i) \)

To introduce a learning component, one may either learn the primal proximal (\( \text{prox}_{\sigma \mathcal{G}} \)) or the dual proximal (\( \text{prox}_{\tau \mathcal{G}} \)), or both. Some recent papers have approached learning the primal proximal operator [11, 42] in the scope of ADMM, but these do not consider learning the dual. It is likely that learning the dual proximal offers an advantage since this allows the inclusion of various operators into the learning. To illustrate this, one can learn a proximal operator for the directional wavelet coefficients of a signal. This is successfully done for denoising [24], and would likely be useful for reconstruction as well.

Learning the dual proximal also allows one to incorporate memory into the algorithm. This can be done for the above case by extending the operator \( K \) so that it also contains a zero component:

\[
K: X \to Y \times X^d \times X^M \quad \text{with} \quad K(f) := \left[ \mathcal{F}(f), \nabla f, 0 \right].
\]
We intend to further elaborate on this approach in an upcoming paper, at this stage we settle with providing an example reconstruction shown in figure 7.

5.2. Choice of discretisation

The given examples were performed using the simplest discretisation of the space \( X \), a pixel basis, but algorithm 2 also works with other representations such as Fourier, wavelet or shearlet coefficients. This could in many cases be better suited for the inverse problem in question and especially so if the operator \( T \) or the regulariser \( S \) has a simple form in this representation, as with the Fourier transform for MRI.

5.3. Choice of error functional

We simply investigated the squared norm error function, \( E(f) = \| f - f_{true} \|_X^2 \), but experience tells us that this is perhaps not the best predictor of human observer performance on a given image. For example, the given algorithm gives a 5 dB improvement over the TV algorithm, but by visual inspection the improvement is not equally drastic. A possible way to improve this and further leverage the power of the learning approach is to use a more sophisticated error functional. Here, performing end-to-end optimisation should be a feasible alternative, s, instead maximise some type of task based measure.

Acknowledgments

The work of both authors was supported by the Swedish Foundation of Strategic Research grant AM13-0049 and Industrial PhD grant ID14-0055. Jonas Adler’s work was also supported by Elekta.

ORCID iDs

Jonas Adler @ https://orcid.org/0000-0001-9928-3407
Ozan Öktem @ https://orcid.org/0000-0002-1118-6483
References


[17] Fidler T, Grasmair M and Scherzer O 2012 Shape reconstruction with a priori knowledge based on integral invariants SIAM J. Imaging Sci. 5 739–45


24


[28] Louis A K 2011 Feature reconstruction in inverse problems Inverse Problems 27 065010


[38] Vakhania N N, Tarieladze V I and Chobanyan S A 1987 Probability Distributions on Banach Spaces (Mathematics and Its Applications (Soviet Series)) (Dordrecht: Kluwer)


Learned Primal-Dual Reconstruction
Jonas Adler and Ozan Öktem

Abstract—We propose the Learned Primal-Dual algorithm for tomographic reconstruction. The algorithm accounts for a (possibly non-linear) forward operator in a deep neural network by unrolling a proximal primal-dual optimization method, but where the proximal operators have been replaced with convolutional neural networks. The algorithm is trained end-to-end, working directly from raw measured data and it does not depend on any initial reconstruction such as filtered back-projection (FBP). We compare performance of the proposed method on low dose computed tomography reconstruction against FBP, total variation (TV), and deep learning based post-processing of FBP. For the Shepp-Logan phantom we obtain >6 dB peak signal to noise ratio improvement against all compared methods. For human phantoms the corresponding improvement is 6.6 dB over TV and 2.2 dB over learned post-processing along with a substantial improvement in the structural similarity index. Finally, our algorithm involves only ten forward-back-projection computations, making the method feasible for time critical clinical applications.

Index Terms—Inverse problems, tomography, deep learning, primal-dual, optimization.

I. INTRODUCTION

In an inverse problem, the goal is to reconstruct parameters characterizing the system under investigation from indirect observations. Such problems arise in several areas of science and engineering, like tomographic imaging where one seeks to visualize the interior structure of an object (2D/3D image) from indirect observations. Imaging technologies of this type, such as computed tomography (CT) and magnetic resonance imaging (MRI) imaging, are indispensable for contemporary medical diagnostics, intervention and monitoring. There is by now a rich theory for model driven tomographic image reconstruction. A key component is knowledge about the forward model, which describes the data formation process in absence of noise. In many applications, an explicit forward model can be derived starting from the underlying physical principles that are utilized by the imaging modality. Another component accounts for the knowledge about the statistical properties of data and a priori information about the image to be recovered.

A parallel line of development in signal processing has been the usage of deep learning for solving a wide range of tasks that can be cast as supervised learning. The success of these data-driven approaches are thus far confined to tasks where knowledge about the forward model is not needed, or has little importance. As an example, an entirely data driven approach for tomographic image reconstruction applicable to clinical sized problems has yet to be demonstrated.

A central question is whether one can combine elements of model and data driven approaches for solving ill-posed inverse problems. In particular, is there a framework for incorporating the knowledge of a forward model when designing a neural network for reconstruction? The Learned Primal-Dual reconstruction method developed in this paper is such a framework. It applies to general inverse problems and it is best described in an abstract setting, so our starting point is to formalize the notion of an inverse problem.

Mathematically, an inverse problem can be formulated as reconstructing (estimating) a signal \( f_{\text{true}} \in X \) from data \( g \in Y \) where

\[
g = T(f_{\text{true}}) + \delta g.
\]  

(1)

Here, the reconstruction space \( X \) and the data space \( Y \) are typically Hilbert Spaces, \( T : X \to Y \) is the forward operator that models how a signal gives rise to data in absence of noise, and \( \delta g \in Y \) is a single sample of a \( Y \)-valued random variable that represents the noise component of data.

A. Variational Regularization

A common model driven approach for solving (1) is to maximize the likelihood of the signal, or equivalently minimizing the negative data log-likelihood [1]:

\[
\min f \in X \quad \mathbb{E} \left[ L(T(f), g) \right].
\]

(2)

This minimization is for typical choices of \( T \) ill-posed, that is, a solution (if it exists) is unstable with respect to the data \( g \) in the sense that small changes to data results in large changes to a reconstruction. Hence, a maximum likelihood solution typically leads to over-fitting against data.

Variational regularization, also referred to as model based iterative reconstruction in medical image processing, avoids over-fitting by introducing a functional \( S : X \to \mathbb{R} \) (regularization functional) that encodes a priori information...
about the true (unknown) \( f_{\text{true}} \) and penalizes unlikely solutions \([2],[3]\). Hence, instead of minimizing only the negative data log-likelihood as in (2), one now seeks to minimize a regularized objective functional by solving
\[
\min_{f \in X} \left[ L(T(f), g) + \lambda S(f) \right] \quad \text{for a fixed } \lambda \geq 0. \tag{3}
\]
In the above, \( \lambda \) (regularization parameter) governs the influence of the a priori knowledge encoded by the regularization functional against the need to fit data.

B. Machine Learning in Inverse Problems

We here review results on learned iterative schemes, see \([4]\) for a wider review on machine learning for medical imaging and \([5]\) for usage of machine learning for solving inverse problems in general.

Machine learning is widely used for non-linear function approximation under weak assumptions and has recently emerged as the state of the art for several image processing tasks such as classification and segmentation. Applied to the inverse problem in (1), it can be phrased as the problem of finding a (non-linear) mapping \( T^\theta : Y \rightarrow X \) satisfying the following pseudo-inverse property:
\[
T^\theta(g) \approx f_{\text{true}} \quad \text{whenever data } g \text{ is related to } f_{\text{true}} \text{ as in (1)}.
\]

A key element in machine learning approaches is to parameterize the set of such pseudo-inverse operators by a parameter \( \theta \in \Theta \) where \( \Theta \) is some parameter space and the main algorithmic complication is to select an appropriate structure of \( T^\theta \) such that, given appropriate training, the pseudo-inverse property is satisfied as well as possible.

In the context of tomographic reconstruction, three main research directions have been proposed. The first is so called learned post-processing or learned denoisers. Here, the learned reconstruction operator is of the form
\[
T^\theta = \Lambda_0 \circ T^\beta
\]
where \( \Lambda_0 : X \rightarrow X \) is a learned post-processing operator and \( T^\beta : Y \rightarrow X \) is some approximate pseudo-inverse, e.g. given by filtered back-projection (FBP) in CT reconstruction. This type of method is relatively easy to implement, given that the pseudo-inverse can be applied off-line, before the learning is performed, which reduces the learning to inferring an \( X \rightarrow X \) transformation. This has been investigated in \([6],[7],[8]\).

Another method is to learn a regularizer and use this regularizer in a classical variational reconstruction scheme according to (3). Examples of this include dictionary learning \([9]\), but several alternative methods have been investigated, such as learning a variational auto-encoder \([10]\) or using a cascade of wavelet transforms (scattering transform) \([11]\).

Finally, some authors investigate learning the full reconstruction operator, going all the way from data to reconstruction. Doing this in one step is typically very computationally expensive and does not scale to the data sizes encountered in tomographic reconstruction. Instead, \textit{learned iterative schemes} have been studied. These schemes resemble classical optimization methods used for tomographic reconstruction but use machine learning to find the best update in each iteration given the last iterate and results of applying the forward operator and its adjoint as input.

One of the first works on learned iterative schemes is \([12]\), which learns an ADMM-like scheme for MRI reconstruction. A further development along this lines is given in \([13]\), which learns over a broader class of schemes instead of ADMM-type of schemes. The application is to finite dimensional inverse problems typically arising in image restoration. This approach was in \([5]\) further extended to non-linear forward operators in to the infinite dimensional setting, which also applies learned iterative schemes to (non-linear, pre-log) CT.

Similar approaches for MRI reconstruction have also been considered \([14],[15]\). Here, the situation is simpler than CT reconstruction since the forward operator is approximated by a Fourier transform, i.e. MRI reconstruction amounts to inverting the Fourier transform.

Given a structure of \( T^\beta \), the “learning” part refers to choosing an “optimal” set of parameters \( \theta \) given some training data, where the concept of optimality is typically quantified through a loss functional that measures the quality of a learned pseudo-inverse \( T^\theta \).

To define this loss functional, consider a \((X \times Y)\)-valued random variable \((f,g)\) with joint probability distribution \( \mu \). This could be e.g. the probability distribution of human bodies and corresponding noisy tomographic data. We define the optimal reconstruction operator as the one whose reconstructions have the lowest average mean squared distance to the true reconstructions, where the average is taken w.r.t. \( \mu \). Finding this reconstruction operator is then given by selecting the parameters \( \theta \in \Theta \) so that the loss functional \( L(\theta) \) is minimized:
\[
L(\theta) := \mathbb{E}_{(f,g) \sim \mu} \left[ \| T^\theta(g) - f_{\text{true}} \|_X^2 \right]. \tag{4}
\]

However, in practice we often do not have access to the probability distribution \( \mu \) of the random variable \((f,g)\). Instead, we know a finite set of samples \((g_1, f_1), \ldots, (g_N, f_N)\). In this setting, we replace \( \mu \) in (4) with its empirical counterpart, so the loss function is replaced with the empirical loss
\[
\hat{L}(\theta) := \frac{1}{N} \sum_{i=1}^{N} \left[ \| T^\theta(g_i) - f_i \|_X^2 \right]. \tag{5}
\]
Since our main goal is to minimize the loss functional, our practical goal is thus two-fold: we want to find a learned reconstruction scheme that minimizes the empirical loss, and we also want it to generalize to new, unseen data.

II. CONTRIBUTION AND OVERVIEW OF PAPER

This paper proposes the \textit{Learned Primal-Dual} algorithm, a general framework for solving inverse problems that combines deep learning with model based reconstruction. The proposed learned iterative reconstruction scheme involves convolutional neural networks (CNNs) in both the reconstruction and data space, and these are connected by the forward operator and its adjoint. We train the networks to minimize the mean squared error of the reconstruction and demonstrate
that this achieves very high performance in CT reconstruction, surpassing recent learning based methods on both analytical and human data.

We emphasize that we learn the whole reconstruction operator, mapping data to reconstruction, and not just a post-processing nor only the prior in isolation.

In addition, we make all of our code and learned parameters open source so that the community can reproduce the results and apply the methods to other inverse problems [16].

III. THE LEARNED PRIMAL-DUAL ALGORITHM

We here introduce how primal-dual algorithms can be learned from data and how this can be used to solve inverse problems.

A. Primal-Dual Optimization Schemes

In imaging, the minimization in (3) is a large scale optimization problem, which traditionally has been addressed using gradient based methods such as gradient descent or its extensions to higher order derivatives, e.g. quasi-Newton or Newton methods. However, many regularizers of interest result in a non-differentiable objective functional, so gradient based methods are not applicable. A common approach to handle this difficulty is to consider a smooth approximation, which however introduces additional parameters and gives non-exact solutions.

An alternative approach is to use methods from non-smooth convex optimization. Primal methods have been developed in order to directly work with non-smooth objective functionals. Here, a proximal step replaces the gradient step. The simplest example of such an algorithm is the proximal point algorithm for minimizing an objective functional \( G \): \( X \rightarrow \mathbb{R} \). It can be seen as the proximal equivalent of the gradient descent scheme and is given by

\[
 f_{i+1} = \text{prox}_{\gamma G}(f_i)
\]

where \( \tau \in \mathbb{R}^+ \) is a step size and the proximal operator is defined by

\[
 \text{prox}_{\gamma G}(f) = \arg \min_{f' \in X} \left[ G(f') + \frac{1}{2\tau} \| f' - f \|_2^2 \right] \tag{7}
\]

While this algorithm could, in theory, be applied to solve (3) it is rarely used directly since (7) does not have a closed form solution. Proximal primal-dual schemes offer a workaround. In these schemes, an auxiliary dual variable in the range of the operator is introduced and the primal \((f \in X)\) and dual variables are updated in an alternating manner.

One well known primal-dual scheme is the primal dual hybrid gradient (PDHG) algorithm [17], also known as the Chambolle-Pock algorithm, with a recent extension to non-linear operators [18]. The scheme (algorithm 1) is adapted for minimization problems with the following structure:

\[
 \min_{f \in X} \left[ F(K(f)) + G(f) \right] \tag{8}
\]

where \( K : X \rightarrow U \) is a (possibly non-linear) operator, \( U \) is a Hilbert space and \( F : U \rightarrow \mathbb{R} \) and \( G : X \rightarrow \mathbb{R} \) are functionals on the dual/primal spaces. Note that (3) is a special case of (8) if we set \( F := \mathcal{L}(\cdot, g) \), \( K := T \) and \( G := S \).

In algorithm 1, \( F^* \) denotes the Fenchel conjugate of \( F \), \( h \in U \) is the dual variable and \([0, K(f_i)]^* : U \rightarrow X \) is the adjoint of the Fréchet derivative of \( K \) in point \( f_i \).

**Example (Total Variation (TV) Regularized CT):** The PDHG method has been widely applied to CT [19]. In CT, the forward operator is given by the ray-transform \( P : X \rightarrow Y \), which integrates the signal over a set of lines \( M \) given by the acquisition geometry. Hence, elements in \( Y \) are functions on lines \( \mathcal{P}(f)(\ell) := \int_{M \cap \ell} f(x) \, dx \) for \( \ell \in M \). and the adjoint of the derivative is the back-projection \( \gamma \). A typical example of variational regularization in imaging is TV regularization, which applies to signals that are represented by scalar functions of bounded variation. The corresponding regularization functional is then given as the 1-norm of the gradient magnitude, i.e. \( S(f) := \| \nabla f \|_1 \), \( \nabla : X \rightarrow \mathbb{R}^d \) is the dimension of the space.

The PDHG method can be used to solve the TV regularized CT optimization problem

\[
 \min_{f \in X} \| \mathcal{P}(f) - g \|_2^2 + \lambda \| \nabla f \|_1.
\]

Since the proximal of \( f \mapsto \| \nabla f \|_1 \) is hard to compute, the following identification is better suited for recasting the above into (8):

\[
 K : X \rightarrow Y \times \mathbb{R}^d \text{ as } K(f) := [P(f), \nabla f], \quad F([h^{(1)}, h^{(2)}]) := \| h^{(1)} \|_2^2 + \| h^{(2)} \|_1 \quad \text{and } G(f) := 0.
\]

**B. Learned PDHG**

The aim is to derive a learned reconstruction scheme inspired by PDHG, algorithm 1. We follow the observation in [21] and [22], that proximal operators can be replaced by other operators that are not necessarily proximal operators. The aforementioned publications replace a proximal operator with a denoising operator such as Block Matching 3D (BM3D). Our idea is to replace the proximal operators by *parametrized* operators where the parameters are learned from training data, resulting in a learned reconstruction operator.

In order to make the learned reconstruction operator well defined and implementable on a computer we also need to select a stopping criterion. Choosing a proper stopping criterion is an active research area, but for simplicity and usability we use a fixed number of iterates. By selecting a fixed number of iterations, the computation budget is also fixed.
prior to training, which is a highly desirable property in time
critical applications.

Algorithm 2 below outlines the resulting variant of the
PDHG algorithm with $i$ iterations in which the primal proxi-
mal has been replaced by a learned proximal, $\hat{\Gamma}_{\theta}$ and the
dual proximal by a learned proximal $\hat{\Lambda}_{\phi}$. Note that in this article
we consider only a single forward model and no regularizing
operator, so we have $K = T$, $U = Y$, but we give the
algorithm in full generality for completeness.

Algorithm 2 Learned PDHG
1: Initialize $f_0 \in X, h_0 \in U$
2: for $i = 1, \ldots, I$ do
3: $h_{i+1} = \hat{\Gamma}_{\phi}(h_i - \tau\partial K(f_i))$
4: $f_{i+1} = f_i - \alpha(g_1(f_i) + \partial K(f_i)e(h_i))$
5: return $f_I$

In algorithm 2, there are several parameters that need to be
selected. These are the parameters of the dual proximal, $\hat{\phi}$,
the primal proximal, $\hat{\theta}$, the step lengths, $\alpha$, $\tau$ and the
overrelaxation parameter, $\theta$. In a learned PDHG algorithm
these would all be inferred, learned, from training data.

We implemented this algorithm and show its performance in
the results section. While the performance was comparable
to traditional methods, it did not improve upon the state of
the art in deep learning based image reconstruction.

C. Learned Primal-Dual

To gain substantial improvements, guided by recent advances
in machine learning, the following modifications to the
learned PDHG algorithm were done.

- Following [5], [13], extend the primal space to allow the
  algorithm some “memory” between the iterations. $f = \{f^{(1)}, f^{(2)}, \ldots, f^{(N_{\text{primal}})}\} \in X^{N_{\text{primal}}}$
- Similarly extend the dual space $U$ to $U^{N_{\text{dual}}}$. For the dual
  instead of explicitly enforcing updates of the form $h_i +
  \tau\partial K(f_i)$, allow the network to learn how to combine
  the previous update with the result of the operator evaluation.
  Instead of hard-coding the over-relaxation $f_{i+1} \leftarrow f_i +
  \theta(f_{i+1} - f_i)$, let the network freely learn in what point
  the forward operator should be evaluated.

The above modifications result in a new algorithm, hence-
forth called the Learned Primal-Dual algorithm, that is out-
lined in algorithm 3.

1) Choice of Starting Point: In theory, the Learned Primal-

Algorithm 3 Learned Primal-Dual
1: Initialize $f_0 \in X^{N_{\text{primal}}}, h_0 \in U^{N_{\text{dual}}}$
2: for $i = 1, \ldots, I$ do
3: $h_i = \hat{\Gamma}_{\phi}(h_{i-1}, K(f_i^{(2)}))$
4: $f_i \leftarrow \hat{\Lambda}_{\phi}(f_{i-1}, \partial K(f_i^{(1)}))$
5: return $f_I$

In cases where a good starting guess is available, it would
make sense to use it. One such option is to assume that there
exists a pseudo-inverse $T'$, e.g. FBP for CT. For the dual
variable, the data $g$ enters into each iterate so there is no need
for a good initial guess. This gives the starting point

$$f_0 = \{T'(g), T'(g), \ldots, T'(g)\}$$
$$h_0 = \{0, 0, \ldots, 0\}$$

In our tests we found that providing the Learned Primal-
Dual algorithm with such an initial guess marginally decreased
training time, but did not give better final results. Given that
using the pseudo-inverse $T'$ adds more complexity by mak-
ing the learned reconstruction operator depend on an earlier
reconstruction, we report values only from zero-initialization.

D. Connection to Variational Regularization

We note that by selecting $N_{\text{primal}} = 2$ and $N_{\text{dual}} = 1$
the Learned Primal-Dual algorithm naturally reduces to the
classical PDHG algorithm by making the following choices:

$$\Gamma_{\phi}(h, K(f^{(2)}), g) = \text{prox}_{\phi}(h + \sigma K(f^{(2)}))$$
$$\Lambda_{\phi}(f^{(1)}, \partial K(f^{(1)})) = \text{prox}_{\phi}(f^{(1)} - \tau\partial K(f^{(1)}))$$

Even if the learned proximal operators do not have explicit
access to the proximals, the universal approximation property
of neural networks [23] guarantees that given sufficient train-
ing data these expressions can be approximated arbitrarily well.

A wide range of other optimization schemes can also be
seen as special cases of the Learned Primal-Dual algorithm.
For example, the gradient descent algorithm with step-length
$\alpha$ for solving (8) is given by

$$f_{i+1} = f_i - \alpha(\partial K(f_i)) + \partial K(f_i)e(h_i))$$

and can be obtained by selecting

$$\Gamma_{\phi}(h, K(f^{(2)}), g) = \text{prox}_{\phi}(K(f^{(2)}))$$
$$\Lambda_{\phi}(f^{(1)}, \partial K(f^{(1)})) = \text{prox}_{\phi}(f^{(1)} - \tau\partial K(f^{(1)}))$$
More advanced gradient based methods such as Limited memory BFGS are likewise sub-cases obtained by appropriate choices of learned proximal operators.

In summary, the Learned Primal-Dual algorithm contains a wide range of optimization schemes as special cases. If the parameters are appropriately selected, then the proposed algorithm should always perform at least as well as current variational regularization schemes given the same stopping criteria, which here is a fixed number of iterates.

IV. IMPLEMENTATION AND EVALUATION

We evaluate the algorithm on two low dose CT problems. One simplified using analytical phantoms based on ellipses and one with a more realistic forward model and human phantoms. We briefly describe these test cases and how we implemented the Learned Primal-Dual algorithm. We also describe the methods we compare against.

A. Test Cases

1) Ellipse Phantoms: This problem is identical to [5] and we restate it briefly. Training data is randomly generated ellipses on a 128 \times 128 pixel domain. The forward operator is the ray transform and hence \( T = P \).

The projection geometry was a sparse 30 view parallel beam geometry with 182 detector pixels. 5\% additive Gaussian noise was added to the projections. Since the forward operator is linear, the adjoint of the derivative is simply the adjoint, which for the ray transform is the back-projection

\[
[\partial T(f)]^\ast = P^\ast.
\]

2) Human Phantoms: In order to evaluate the algorithm on a clinically realistic use-case we consider reconstruction of simulated data from human abdomen CT scans as provided by Mayo Clinic for the AAPM Low Dose CT Grand Challenge [24]. The data includes full dose CT scans from 10 patients, of which we used 9 for training and 1 for evaluation. We used the 3 mm slice thickness reconstructions, resulting in 2168 training images, each 512 \times 512 pixel in size. Thus, given that we minimize the pointwise error, the total number of data-points is \( 512^2 \times 2168 \approx 5 \times 10^8 \).

We used a two-dimensional fan-beam geometry with 1000 angles, 1000 pixels, source to axis distance 500 mm and axis to detector distance 500 mm. In this setting, we consider the more physically correct non-linear forward model given by Beer-Lamberts law

\[
T(f)(\xi) = e^{-\mu P(f)(\xi)}
\]

where the unit of \( f \) is g/cm\(^3\) and \( \mu \) is the mass attenuation coefficient, in this work selected to 0.2 cm\(^2\)/g which is approximately the value in water at x-ray energies. We used Poisson noise corresponding to \( 10^4 \) incident photons per pixel before attenuation, which would correspond to a low dose CT scan. We find the action of the adjoint of the derivative by straightforward computation

\[
[\partial T(f)]^\ast(g) = -\mu P^\ast(e^{-\mu P(f)} g)(\cdot)
\]

for \( g \in Y \).

For validation of the ellipse data case, we simply use the (modified) Shepp-Logan phantom and for the human phantom data we use one held out set of patient data. See fig. 1 for examples.

B. Implementation

The methods described above were implemented in Python using Operator Discretization Library (ODL) [25] and TensorFlow [26]. All operator-related components, such as the forward operator \( T \), were implemented in ODL, and these were then converted into TensorFlow layers using the as_tensorflow_layer functionality of ODL. The neural network layers and training were implemented using TensorFlow. The implementation utilizes abstract ODL structures.
for representing functional analytic notions and is therefore
generic and easily adaptable to other modalities.

We used the ODL operator RayTransform in order to
evaluate the ray transform and its adjoint using the
graphics processing unit (GPU) accelerated 'astra_gpu'
backend [27].

1) Deep Neural Network and Training Details: Given the
general Learned Primal-Dual scheme in algorithm 3, a parametriza-
tion of the learned proximal operators is needed in order
to proceed. In many inverse problems, and particularly in CT
and MRI reconstruction, most of the useful properties for both
the forward operator and prior are approximately translation
invariant. For this reason the resulting reconstruction operator
should be approximately translation invariant, which indicates
that CNNs are suitable for parametrizing the aforementioned
reconstruction operator.

We used learned proximal operators of the form
\[ \text{Id} + W_{w_j,b_j} \circ A_{\theta} \circ W_{w_j,b_j} \circ A_{\theta} \circ W_{w_j,b_j} \]
where Id is the identity operator that makes the network a residual
network. There are two main reasons for choosing such a structure. First, proximal operators (as the name implies) are typically close to the identity and second, there is rich evidence in the machine learning literature [28] that
CNNs are suitable for parametrizing the aforementioned reconstruction operator.

We used affine operators \( W_{w_j,b_j} \) parametrized
by weights \( w_j \) and biases \( b_j \). The affine operators are defined
in terms of so called convolution operators (here given on the
primal space, but equivalently on the dual space). These are
given as affine combinations of regular convolutions, more specifically:
\[ W_{w_j,b_j} : X^n \rightarrow X^m \]
where the \( k \)-th component is given by
\[ [W_{w_j,b_j}(f_1^{(n)}, \cdots, f_{(n)l})]^{(k)} = b_j^{(k)} + \sum_{l=1}^{n} w_j^{(k)} f^{(l)} \]
where \( b_j \in \mathbb{R}^m \) and \( w_j \in \mathbb{R}^{n \times m} \).

The non-linearities were chosen to be Parametric Rectified
Linear Units (PReLU) functions
\[ A_{\theta}(x) = \begin{cases} x & \text{if } x \geq 0 \\ -c_j x & \text{else.} \end{cases} \]
This type of non-linearity has proven successful in other
applications such as classification [29].

We let the number of data that persists between the iterates be \( N_{\text{data}} = 5 \). The convolutions were all \( 3 \times 3 \) pixel
size, and the number of channels was, for each primal learned
proximal, \( 6 \rightarrow 32 \rightarrow 32 \rightarrow 5 \), and for the duals \( 7 \rightarrow 32 \rightarrow 32 \rightarrow 5 \) where the higher number of inputs is due to the data \( g \)
being supplied to the dual proximals.

We let the number of unrolled iterations be \( I = 10 \), that is the
operator \( T \) and the adjoint of its derivative \( [T(f_1^{(n)l})]^{*} \)
are both evaluated 10 times by the network. Since each iterate
involves two 3-layer networks, one for each primal, the total
depth of the network is 60 convolutional layers and the total
number of parameters approximately \( 2.4 \times 10^7 \). In the context of
deep learning, this is a deep network but with a small number of
parameters. The network is visualized in fig. 2.

We used the Xavier initialization scheme [30] for the
convolution parameters, and initialized all biases to zero.

We trained the network by minimizing the empirical loss (5)
using training data as explained above using the ADAM
optimizer in TensorFlow [31]. We used \( 10^5 \) batches on each
problem and used a learning rate schedule according to cosine
annealing [32], i.e. the learning rate in step \( t \) was
\[ \eta_t = \frac{\eta_0}{2} \left( 1 + \cos \left( \pi \frac{t}{t_{\text{max}}} \right) \right) \]
where the initial learning rate $\eta_0$ was set to $10^{-3}$. We also let the parameter $\beta_0$ of the ADAM optimizer to 0.99 and let all other parameters use the default choices. We performed global gradient norm clipping [33], limiting the gradient norms to 1 in order to improve training stability and used a batch size of 5 for the ellipse data and 1 for the human phantoms.

We did not use any regularization of the learned parameters, nor did we utilize any tricks such as dropout or batch normalization. Neither did we perform any data augmentation.

The training was done using a single GTX 1080 TI GPU and took about 11 hours for the ellipse data and 40 hours for the human phantoms.

2) Incorporating the Forward Operator in Neural Networks: In order to minimize the loss function (4), stochastic gradient descent (SGD) type methods are typically used and these require (an estimate of) the gradient of the loss function $L(\theta) = \mathbb{E}_{d \sim \mathcal{P}_d}[\frac{1}{2}\|\theta \partial_T f_i(\mathbf{g})^* (\gamma g + \tilde{f}_i) - \tilde{f}_i\|^2]$, where $[\theta \partial_T f_i(\mathbf{g})]^*$ is the adjoint of the derivative (with respect to $\theta$) of the learned reconstruction operator applied in $\mathbf{g}$, often called gradient in the machine learning literature. This introduces a challenge since it will depend on each component of the neural network, including the learned proximal operators but also the forward operator $T$ and the backward operator $[\theta \partial T(f)]^*$, propagated through all $I$ iterations.

To solve this, we used the built in automatic differentiation functionality of TensorFlow which uses the chain rule (back-propagation). This in turn requires the adjoints of the derivatives of each individual component which for the proximals were computed by TensorFlow and for the operators by ODL.

C. Comparison

We compare the algorithm to several widely used algorithms, including standard FBP and (isotropic) TV regularized reconstruction. We also compare against several learned schemes. These are briefly summarize here, see the references for full descriptions.

The FBP reconstructions were done with a Hann filter and used the method $\text{DeepS}^{\text{op}}$ in ODL. The TV reconstruction was performed using 1000 iterations of the classical PDHG algorithm, implemented in ODL as $\text{pdhg}$. The filter bandwidth in the FBP reconstruction and the regularization parameter in the TV reconstruction were selected in order to maximize the peak signal to noise ratio (PSNR).

The partially learned Gradient method in [5] is similar to the algorithm proposed in this article, but differs in that instead of learning proximal operators it learns a gradient operator and the forward operator enters into the neural network through the gradient of the data likelihood. Publicly available code and parameters [34] were used.

The next comparison is against a deep learning based approach for post-processing based on a so called U-Net [35]. The U-Net was first proposed for image segmentation, but by changing the number of output channels to one, it can also be used for post-processing as was done in [7]. Here an initial reconstruction is first performed using FBP and a neural network is trained on pairs of noisy FBP images and noiseless/flow noise ground truth images, learning a mapping between them. We re-implemented the algorithm from [7] but found that using the training procedure as stated in the paper gave sub-optimal results. We hence report values from using the same training scheme as for our other algorithms in order to give a more fair comparison.

Additionally, our comparison includes learned PDHG, algorithm 2, as well as the following two simplified versions of the Learned Primal-Dual algorithm. The first is a Learned Primal algorithm, which does not learn any parameters for the dual proximal, instead it returns the residual

$$\Gamma_\theta^I(b_{i-1}, T(f_{i-1}), x) = T(f_{i-1}) - g$$

The second, FBP + residual denoising algorithm, further simplifies the problem by discarding the forward operator completely, and can be seen as selecting

$$\Lambda_\theta^I(f_{i-1}, [\theta \partial_T T(f)]^*(b_{i-1})) = \Lambda_\theta^I(f_{i-1})$$

Since this method does not have access to the data $g$, we select the initial guess according to a FBP, see (10). This makes the algorithm a learned denoiser.

For the human phantoms we compare both non-linear and linearized versions of the forward operator, but given that training times are noticeably longer, we only compare to the previously established methods of FBP, TV and U-Net denoising.

All learned algorithms were trained using the same training scheme as outlined in section IV-B1, and measure the runtime, PSNR and the structural similarity index (SSIM) [36].

All methods that we compare against are available in the accompanying source code.

V. Results

The quantitative results for the ellipse data is given in table I, where we can see that the proposed Learned Primal-Dual algorithm out-performs the classical schemes (FBP and TV) significantly w.r.t. the reconstruction error as measured by both PSNR and SSIM. We also note that the Learned Primal-Dual algorithm gives a large improvement over the previous deep learning based methods such as the learned gradient scheme and U-Net based post-processing, giving an improvement exceeding 6 dB. The Learned Primal-Dual algorithm also outperforms the Learned PDHG and the FBP + residual denoising algorithms by wide margins.

The only method that achieves results close to the Learned Primal-Dual algorithm is the Learned Primal method, but the Learned Primal-Dual algorithm gives a noticeable improvement of 1.3 dB.

The results are visualized in fig. 3. We note that small structures, such as the small inserts, are much more clearly visible in the Learned Primal and Learned Primal-Dual reconstructions than in the other reconstructions. We also note that both the Learned PDHG and Learned Primal reconstruction seem to have a halo artefact close to the outer bone which is absent in the Learned Primal-Dual reconstruction.
Fig. 3. Reconstructions for the ellipse data using the compared methods. The window is set to [0.1, 0.4], corresponding to the soft tissue of the modified Shepp-Logan phantom. (a) FBP. (b) TV. (c) FBP + U-Net denoising. (d) FBP + residual denoising. (e) Learned Gradient. (f) Learned PDHG. (g) Learned Primal. (h) Learned Primal-Dual.

With respect to run-time the learned methods that involve calls to the forward operator (Learned Gradient, PDHG, Primal, Primal-Dual) are slower than the methods that do not (FBP + U-Net denoising. Residual) by a factor ≈ 6. When compared to TV regularized reconstruction all learned methods are at least 2 orders of magnitude faster.

Quantitative results for the human phantoms data are presented in Table II. We note that the FBP reconstruction has a much more competitive image quality than it had for the ellipse data, both quantitatively and visually. It is likely for this reason that the FBP + U-Net denoising performs better than it did on the ellipses, outperforming TV by 4.4 dB. However, if we look at the SSIM we note that this improvement does not translate as well to the structural similarity, where the method is comparable to TV regularization.

Both quantitatively and visually, the linear and non-linear versions of the Learned Primal-Dual algorithm give very similar results. We will focus on the linear version which gave slightly better results.

The Learned Primal-Dual algorithm gives a 10.5 dB improvement over the FBP reconstruction, a 6.6 dB improvement over TV and 2.2 dB over the U-Net denoiser. This is less than for the ellipse data, but still represents a large improvement. On the other hand, while the U-Net denoiser did not improve the SSIM as compared to TV regularization, the Learned Primal-Dual algorithm gives a large improvement.

This improvement is also present in the images when inspected visually in Fig. 4. In particular, we see that some artifacts visible in the FBP reconstruction are still discernible in the U-Net denoiser and TV reconstructions. Examples include streak artifacts, especially around the edges of the phantom and structures spuriously created from noise, such as a line in the muscle above the right bone. These are mostly absent in the Learned Primal-Dual reconstruction. However, we do note that the images do look slightly over-smoothed. Both of these observations become especially apparent if we look at the zoomed in regions, where we note that the Learned Primal-Dual algorithm is able to reconstruct finer detail than the other algorithms, but gives a very smooth texture.

With respect to the run-time, the Learned Primal-Dual is more competitive with the FBP and U-Net denoiser algorithms for full size data than for the ellipse data. This is because the size of the data is much larger, which increases the runtime of the FBP reconstruction, which is also needed to compute the initial guess for the U-Net denoiser. As for the ellipse data, both learned methods outperform TV regularized reconstruction by two orders of magnitude with respect to runtime.

VI. DISCUSSION

The results show that the Learned Primal-Dual algorithm outperforms classical reconstruction algorithm by large margins as measured in both PSNR and SSIM and also improves upon learned post-processing methods for both simplified
Fig. 4. Reconstructions of a human phantom along with two zoomed in regions indicated by small circles. The left zoom-in has a true feature whereas texture in right zoom-in is uniform. The window is set to [−200, 200] HU. Among the methods tested, only the Learned Primal-Dual algorithm correctly recovers these regions. In the others, the true feature in the left zoom-in is indistinguishable from other false features of same size/contrast and right zoom-in has a streak artifact. Note also the clinically feasible runtime of the Learned Primal-Dual algorithm. To summarize, the Learned Primal-Dual algorithm offers performance advantages over other methods that translate into true clinical usefulness.

(a) 512 × 512 pixel human phantom. (b) Filtered back-projection (FBP) PSNR 33.65 dB, SSIM 0.830, 423 ms. (c) Total variation (TV) PSNR 37.48 dB, SSIM 0.946, 64371 ms. (d) FBP + U-Net denoising PSNR 41.92 dB, SSIM 0.941, 463 ms. (e) Primal-Dual, linear PSNR 44.10 dB, SSIM 0.969, 620 ms. (f) Primal-Dual, non-linear PSNR 43.91 dB, SSIM 0.969, 670 ms.

Table II: Comparison of the Learned Primal-Dual algorithm with other methods for the human phantom data. Units for entries are the same as in Table I.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>Runtime</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBP</td>
<td>33.65</td>
<td>0.830</td>
<td>423</td>
<td>1</td>
</tr>
<tr>
<td>TV</td>
<td>37.48</td>
<td>0.946</td>
<td>64371</td>
<td>1</td>
</tr>
<tr>
<td>FBP + U-Net denoising</td>
<td>41.92</td>
<td>0.941</td>
<td>463</td>
<td>10^7</td>
</tr>
<tr>
<td>Learned Primal-Dual, linear</td>
<td>44.21</td>
<td>0.969</td>
<td>620</td>
<td>2.4 10^7</td>
</tr>
<tr>
<td>Learned Primal-Dual, non-linear</td>
<td>43.91</td>
<td>0.969</td>
<td>670</td>
<td>2.4 10^7</td>
</tr>
</tbody>
</table>

ellipse data and for human phantoms. In addition, especially for the 512 × 512 human phantoms, the reconstruction time is comparable with even filtered back-projection and learned post-processing.

One interesting, and to the best of our knowledge, unique feature of the Learned Primal-Dual algorithm in the field of deep learning based CT reconstruction, is that it gives reconstructions working directly from data, without any initial reconstruction as input.

Since the algorithm is iterative, we can visualize the iterates to gain insight into how it works. In fig. 5 we show some iterates with the non-linear forward operator. We note that the reconstruction stays very bad until the 8th iterate when most structures seem to come in place, but the image is still noisy. Between the 8th and 10th iterate, we see that the algorithm seems to perform an edge-enhancing step. It thus seems like the learned iterative scheme works in two steps, first finding the large scale structures and then fine-tuning the details.

Similarly to the edge-enhancement that seems to be performed in the primal space, we note that in the dual space the sinogram that is back-projected seems to be band-pass filtered to exclude both very low and very high frequencies.

We note that in the very noisy and under-sampled data used for the ellipse phantoms, the learned algorithms that make use of the forward operator, such as the Learned Gradient, Primal and Primal-Dual algorithms outperform even state of the art post-processing methods by large margins and that in this regimen, TV regularization performs relatively well when compared to post-processing methods. This improvement in reconstruction quality when incorporating the forward operator, while still substantial, is not as large for the human phantom in which the data was less noisy.
Fig. 5. Iterates 2, 4, 6, 8 and 10 in the Learned Primal-Dual algorithm when reconstructing the human phantoms using a non-linear forward model. Left: Reconstruction \( (f_i) \). Middle: Point of evaluation for the forward operator \( (f_i) \). Right: Point of evaluation for the adjoint of the derivative \( (h_i) \). Windows selected to cover most of the range of the values.

To explain this, we conjecture that in the limit of highly noisy data where the initial reconstruction as given by e.g. FBP becomes very bad, learned schemes that incorporate the forward model and work directly from data, such as the Learned Primal-Dual algorithm, has a considerable advantage over post-processing methods and that this advantage increases with decreasing data quality.

Further along these lines, note that for the human data the post-processing gives a large improvement in PSNR when compared to TV regularization, which is not necessarily reflected in the SSIM. On the other hand, the Learned Primal-Dual algorithm improves upon both PSNR and SSIM. This can be explained by the learned post-processing being limited by the information content of the FBP while the Learned Primal-Dual algorithm works directly with data and is thus limited by the information content of the data, which is greater or equal to that of the FBP. In theory, the Learned Primal-Dual algorithm can thus find structures that are not present in the FBP, something post-processing methods cannot.

In these experiments we found that while the algorithm seems to handle non-linear forward models well, we did not observe any notable performance improvement by doing so. This may indicate that performing reconstructions on post-log data is preferable.

The structure of the neural network was not extensively fine-tuned and we suspect that better results could be obtained by a better choice of network for the learned proximal operators. We also observed that the choice of optimizer and learning rate decay had a large impact on the results, and we suspect that further research into how to correctly train learned reconstruction operators will prove fruitful.

Finally, we observe that the reconstructions, while outperforming all of the compared methods with respect to PSNR and SSIM, suffers from a perceived over-smoothing when inspected visually. We suspect that the particular choice of objective function used in this article, the squared norm (4), is a main cause of this and invite future researchers to implement learned reconstruction operators that use more advanced loss functions such as perceptual losses [37].

VII. CONCLUSION

We have proposed a new algorithm in the family of deep learning based iterative reconstruction schemes. The algorithm is inspired by the PDHG algorithm, where we replace the proximal operators by learned operators. In contrast to several recently proposed algorithms, the new algorithm works directly from tomographic data and does not depend on any initial reconstruction.

We showed that the algorithm gives state of the art results on a computed tomography problem for both analytical and human phantoms. For analytical phantoms, it improves upon both classical algorithms such as FBP and TV, and post-processing based algorithms by at least 6 dB while also improving the SSIM. The improvements for the human phantom were more modest, but the algorithm still improves upon a TV regularized reconstruction by 6.6 dB and gives an improvement of 2.2 dB when compared to a learned post-processing.

We hope that this algorithm will inspire further research in Learned Primal-Dual schemes and that the method will be applied to other imaging modalities.

ACKNOWLEDGEMENTS

The authors also thank Dr. Cynthia McCollough, the Mayo Clinic, and the American Association of Physicists in Medicine for providing the data necessary for performing comparison using a human phantom.

REFERENCES


Task adapted reconstruction for inverse problems

Jonas Adler
KTH
Elekta

Sebastian Lunz
University of Cambridge

Olivier Verdier
KTH
Western Norway University of Applied Sciences

Carola-Bibiane Schönlieb
University of Cambridge

Ozan Öktem
KTH


Abstract
The paper considers the problem of performing a task defined on a model parameter that is only observed indirectly through noisy data in an ill-posed inverse problem. A key aspect is to formalize the steps of reconstruction and task as appropriate estimators (non-randomized decision rules) in statistical estimation problems. The implementation makes use of (deep) neural networks to provide a differentiable parametrization of the family of estimators for both steps. These networks are combined and jointly trained against suitable supervised training data in order to minimize a joint differentiable loss function, resulting in an end-to-end task adapted reconstruction method. The suggested framework is generic, yet adaptable, with a plug-and-play structure for adjusting both the inverse problem and the task at hand. More precisely, the data model (forward operator and statistical model of the noise) associated with the inverse problem is exchangeable, e.g., by using neural network architecture given by a learned iterative method. Furthermore, any task that is encodable as a trainable neural network can be used. The approach is demonstrated on joint tomographic image reconstruction, classification and joint tomographic image reconstruction segmentation.
1 Introduction

The overall goal in inverse problems is to determine model parameters such that model predictions match measured data to sufficient accuracy. Such problems arise in various scientific disciplines. One example is biomedical imaging where the image is the “model parameter” that needs to be determined from data acquired using an imaging device like a tomographic scanner or a microscope. The prime example of this is tomographic imaging in medicine which has revolutionized health care over the past 30 years, allowing doctors to find disease earlier and improve patient outcomes [19, 82]. Likewise, scientific computing is nowadays considered to be the “third pillar of science” standing right next to theoretical analysis and experiments for scientific discovery, much thanks to possibilities for simulating and optimizing complex physical and engineering systems. A key element in realizing this role is the ability to solve the inverse problem of calibrating parameters in a mathematical model of the system so that simulations match benchmark data [8].

The inverse problem of reconstructing the model parameter from data is often only one out of many steps in a procedure where the recovered model parameter is used in decision making. The reconstructed model parameter is typically summarized, either by an expert or automatically, and resulting task dependent descriptors are then used as basis for decision making, see fig. 1.

Clearly, there are several disadvantages with performing the various parts of the above pipeline independently from each other. Each single step is prone to introduce approximations that are not accounted for by subsequent steps, the reconstruction may not consider the end task, and the feature extraction may not consider measured data. In fact, the task is almost always only accounted for at the very final step. It is therefore natural to ask whether one may adapt the reconstruction method for the specific task at hand. Task adapted reconstruction refers to methods that integrate the reconstruction procedure with (parts of) the decision making procedure associated with the task. This is sometimes also referred to as “end-to-end” reconstruction.

2 Overview

We start with a brief survey of existing approaches to task adapted reconstruction in the context of tomographic image reconstruction (Section 4), which also points out the drawbacks that come with these approaches. The section that follows (Section 5) introduces the statistical view on inverse problems. More specifically, we consider Bayesian inversion (section 5.1) in which a reconstruction method is a statistical estimator (section 5.2). After pointing out some key challenges associated
Fig. 1: Typical workflow involving an inverse problem. The second row represents the data acquisition where raw data is acquired and pre-processed, resulting in cleaned data. In the third row, the cleaned data is used as input to a reconstruction step that recovers the model parameter, which is then post-processed to extract features that are used as input for model building. The final outcome is a task adapted model that can be used for decision making. The dotted rectangular part outlines the steps that are unified by task adapted reconstruction.

with Bayesian inversion (section 5.3), we introduce the learned iterative methods (section 5.4) that are later used in our applications of task adapted reconstruction (section 7.3). We then switch gears and consider tasks on the model parameter space that can be formulated as a statistical estimation problem (section 6). The first step is to provide an abstract framework (section 6.1) with a plug-and-play structure for adapting to a specific task. To further illustrate the wide applicability of this framework, section 6.2 describes a number of tasks that are worked out in detail followed by further examples in section 6.3.

Section 7 introduces task adapted reconstruction in an abstract setting (section 7.1). It assumes that both the reconstruction (section 5.2) and task (section 6.1) are given by appropriate decision rules. An important part is the computational im-
3 Specific contributions

The paper offers a generic, yet highly adaptable, framework for task adapted reconstruction that is based on considering both the reconstruction and the task as statistical estimation problems. The implementation uses neural networks for both these steps, which is essential for both performance in terms of quality and computational feasibility as shown in the example for joint tomographic reconstruction and segmentation. Both networks are trained jointly using a joint loss function (20) that "interpolates" between sequential and end-to-end approaches. Here, sequential refers to a setting where the neural network for reconstruction is trained separately and its output is used in the training of the neural network for the task. End-to-end is when a neural network for the task is trained directly against data without explicitly introducing a reconstruction step.

To the best of our knowledge, this is the first paper that offers an approach to task adapted reconstruction that unifies reconstruction with such a diverse set of tasks in a computationally feasible manner under the guiding principles of statistical decision theory, learning, and efficient inference algorithms. This allows for re-using algorithmic components thereby opening up new ways of thinking about machine learning and inverse problems that may ultimately lead to deeper understanding of the possibilities for integrating elements of decision making into the reconstruction. Furthermore, introducing the joint loss in (20) and investigating its properties (section 8) are novel contributions. Our work also leaves many open questions and future research directions for the inverse problems and machine learning communities as outlined in section 9.

4 Survey of task adapted tomographic reconstruction

There is an ongoing effort within the inverse problems community to include signal processing steps associated with performing a task jointly with the reconstruction step. In tomographic imaging, most of the tasks considered this far correspond to feature extraction, e.g., segmentation or extraction of features expressed through sparse representations as in compressed sensing. In such case, task adapted reconstruction reduces to joint reconstruction and feature extraction.

Current approaches to task adapted reconstruction are primarily based one the classical approach to inverse problems. In this setting, the problem is to recover the
true unknown feature \( d^* \in D \) from data \( y \in Y \) by solving an operator equation:

\[
y = A(x^*) + e \quad \text{and} \quad d^* = T(x^*).
\]

(1)

The forward operator \( A: X \to Y \) models how a model parameter (image in tomographic imaging) gives rise to data and the task operator \( T: X \to D \) represents the feature extraction. In the above, both these are assumed to be known. Likewise, \( e \) is generated by a \( Y \)-valued random variable \( e \sim \mathcal{P}_{\text{noise}} \) with known distribution, so task adapted reconstruction reduces to finding the dotted operator in (2).

\[
X \xrightarrow{T} D \xrightarrow{A} Y
\]

(2)

Note that the task operator \( T: X \to D \) is often highly non-injective, so it makes no sense to consider \( A \circ T^{-1}: D \to Y \) as “new forward operator”.

Approaches based on “solving” (1) heavily depend on the nature of the \( T \) and below is a brief list of prior work in the context of tomographic imaging.

**Edge recovery:** Lambda-tomography [53] is a non-iterative method that recovers edges directly from noisy tomographic data using the canonical relation from microlocal analysis. Another non-iterative approach combines the method of approximate inverse with an explicit task operator, e.g., a Canny edge detector [62]. Finally, it is also possible to use a variational approach with suitable regularizer. Examples relevant for edge recovery are variants of total variation (TV) [11, 7] or sparsity promoting \( \ell_1 \)-type of regularizers with an underlying dictionary that is specifically designed to sparsely represent edges, like curvelets, shearlets, beamlets, and bandlets [27, 83, 55].

**Segmentation:** Methods for joint reconstruction and segmentation is an active area of research. Most approaches are based on a variational scheme with suitably chosen regularizers and control variables. One example is usage of Mumford-Shah penalty [78, 44], another is based on level set approaches [99]. A further refinement is to consider semantic segmentation. Here, a variational scheme that amounts to computing a maximum a posteriori (MAP) estimator with a Gauss-Markov-Potts type of prior shows promising results on small-scale examples [70, 80].

**Image registration:** To register a template against an indirectly observed target (indirect image registration) is a key step for reconstruction in spatiotemporal imaging. The (temporal) deformation can be modeled using optical flow [10] or diffeomorphic deformations [14, 32]. Yet another approach is to consider optimal transport [50].

Approaches to task adapted reconstruction that are based on solving (1) suffer from two issues that seriously limit their usefulness in practical imaging applications.
The first is the requirement for an explicit handcrafted task operator $T: X \to D$. More advanced tasks, like many mentioned in sections 6.2 and 6.3, are difficult to encode in this way and available examples mostly consider task operators that extract “simple” features that must be further processed before they can be used for decision making. This is also the reason for why the term “feature reconstruction” [62] is used as a proxy for task adapted reconstruction. The second issue relates to computational feasibility. Evaluating the task operator, like in segmentation and image registration, is computationally demanding and requires setting values for extra (nuisance) parameters. Furthermore, most state-of-the-art approaches for solving (1) are based on variational methods, which quickly become computationally unfeasible for large scale imaging problems.

Many complex tasks have been successfully addressed using techniques from machine learning, so it makes sense to investigate whether such techniques can be integrated with reconstruction for task adapted reconstruction. One example is given in [96] for abnormality (tumor) detection in low-dose computed tomography (CT) imaging. The idea here is to jointly train a learned iterative scheme for reconstruction [2] with a 3D convolutional neural network for detecting the abnormality in the reconstructed images. Another examples introduces a unified deep neural network architecture (SegNetMRI) for combined Fourier inversion (MRI image reconstruction) and segmentation [89]. Here, one has two neural networks with the same encoder-decoder structure, one for MRI reconstruction consisting of multiple cascaded blocks, each containing an encoder-decoder unit and a data fidelity unit, and the other for segmentation. These are pre-trained and coupled by ensuring they share reconstruction encoders. These two examples are special cases of the generic approach develop in section 7.

5 Statistical inverse problems

Let $X$ and $Y$ denote separable Banach spaces where $(X, \mathcal{S}_X)$ and $(Y, \mathcal{S}_Y)$ are measurable spaces. Next, let $\mathcal{P}_X$ and $\mathcal{P}_Y$ denote spaces of probability measures on $X$ and $Y$, respectively. Following [25], a (statistical) inverse problem amounts to reconstructing (estimating) $x^\ast \in X$ from measured data $y \in Y$ that is generated by a $Y$-valued random variable $y$ where

$$y \sim \mathcal{M}(x^\ast) \quad \text{with known } \mathcal{M}: X \to \mathcal{P}_Y \quad \text{(data model)}. \quad (3)$$

Elements in $X$ (model parameter space) represent possible model parameters and elements in $Y$ (data space) represent possible data. In tomographic imaging, elements in $X$ are often functions defined on a fixed domain in $\mathbb{R}^n$ representing images and elements in $Y$ are real-valued functions defined on a fixed manifold $\mathcal{M}$, which is given by the acquisition geometry associated with the measurements. Furthermore, just as in the classical setting, most statistical inverse problems do not have a unique solution in the sense that the model parameter is not identifiable [25, section 2.3].
A common data model is when data is contaminated with additive noise:

\[ y = A(x^\ast) + e \quad \text{with } e \sim P_{\text{noise}} \quad \text{for some known } P_{\text{noise}} \in \mathcal{P}_Y. \]  

(4)

Here, \( A : X \rightarrow Y \) (forward operator) models how data is generated in absence of noise and \( e \sim P_{\text{noise}} \) models noise. If \( e \) is independent from \( x^\ast \), then (4) amounts to the data model

\[ M(x) = \delta_{A(x)} \otimes P_{\text{noise}}(\cdot - A(x)) \quad \text{for any } x \in X. \]

Another data model is when \( M(x) \) is a Poisson random measure on \( Y \) with mean \( A(x) \). This is a suitable data model for imaging modalities that rely on counting statistics in a low-dose setting, such as line of response PET [47] [72, section 3.2] and variants of fluorescence microscopy [41, 21], see also [43, 87] for a more abstract treatment.

### 5.1 Bayesian inversion

Only seeking an estimate of \( x^\ast \in X \) is limiting since it does not account for the uncertainty. A more comprehensive analysis is based on introducing a \( X \)-valued random variable \( x \sim \pi^\ast \) whose true (unknown) probability distribution \( \pi^\ast \in \mathcal{P}_X \) generates \( x^\ast \). One can then rephrase the inverse problem stated earlier as the task of recovering the probability measure \( \pi^\ast \in \mathcal{P}_X \) given data \( y \in Y \) generated by \( y \), which is related to \( x^\ast \) through the data model as in (3). An important special case is when \( \pi^\ast \) is parametrized by \( x^\ast \in X \) in a known way, so the inverse problem reduces to the task of recovering \( x^\ast \in X \).

In a Bayesian setting, one considers the posterior distribution of \( x \) given \( y = y \) up to a constant of proportionality. More precisely, consider a setting where the joint law \((x, y) \sim \mu\) can be written in terms of conditional probabilities:

\[ \mu = \pi_0(x^\ast) \otimes \pi(y | x = x^\ast) = \pi_0(x^\ast) \otimes M(x^\ast). \]  

(5)

Here, \( \pi_0 \) serves as a (possibly improper) prior and the last equality in (5) follows from the definition of the data model as the conditional distribution of \( y \) given \( x = x^\ast \). In particular, the joint law \( \mu \) in (5) is proportional to the posterior, so the decomposition above exists as soon as Bayes’ theorem holds. This is the case in a rather general setting [18, Theorem 14], but a decomposition is also possible in some cases where the prior is not proper\(^1\).

A key point in the Bayesian setting is to explore the posterior distribution of \( x \) given \( y = y \) assuming that both \( x \mapsto \pi_0 \in \mathcal{P}_X \) (prior) and \( x \mapsto M(x) \) (data model) are known, but \( x^\ast \in X \) is unknown. The data model often has an associate

---

\(^1\) Under certain circumstances it is possible to work with improper priors on the model parameter space, e.g., by computing posterior distributions that approximate the posteriors one would have obtained using proper conjugate priors whose extreme values coincide with the improper prior.
density $\mathcal{L}$ (data likelihood) that is known to sufficient degree of accuracy, in which case $d\mathcal{M}(x)(y) = \mathcal{L}(y \mid x) \, dy$.

### 5.2 Reconstruction as an optimal decision rule

A reconstruction method is formally a measurable $X$-valued mapping on $Y$, which in the statistical setting corresponds to an estimator. More precisely, $((Y, \mathcal{S}_Y), \{\mathcal{M}(x)\}_{x \in X})$ defines a statistical model parametrized by the model parameter space $X$ and a reconstruction method corresponds to a point estimator. The latter is a non-randomized decision rule for a statistical estimation problem where the model parameter space $X$ parametrizes the underlying statistical model and at the same time constitutes the decision space. The reader may here consult [60, section 3.1] for formal definitions of decision theoretic notions used here.

There are many possible reconstruction methods (estimators) so one needs a framework where these can be compared against each other. Statistical decision theory offers such a framework by associating a notion of risk to a decision rule. This quantifies the downside that comes with using a particular reconstruction method. The first step is to define the loss function on the decision space, which in our specific setting becomes a measurable mapping (see [60, Definition 3.2] for the definition in a general setting):

$$\ell_X : X \times X \rightarrow \mathbb{R}.$$  

(6)

A common choice in imaging inverse problems is the $L^2$-loss, which is the squared $L^2$-distance. There are however alternatives that are not based on point-wise differences but on differences between high-level image features, e.g., the Wasserstein distance [3] and perceptual losses [46].

Having selected a loss function as in (6) and a prior $\pi_0$ in (5) on the model parameter space, the $\pi_0$-average risk (Bayes risk or expected loss) for reconstruction is given as

$$\mathcal{R}_{\pi_0}(A^\dagger) = \mathbb{E}_{\pi_0 \otimes \mathcal{M}(x)} [\ell_X (x, A^\dagger(y))] .$$

(7)

A natural criteria to select a reconstruction method (estimator) is to minimize Bayes risk, i.e., to select an estimator (non-randomized decision rule) that minimizes $A^\dagger \mapsto \mathcal{R}_{\pi_0}(A^\dagger)$ in (7).

Note here that in the finite dimensional setting, minimizing Bayes risk is the same as computing the conditional mean (posterior mean) if and only if the loss function in (6) is the Bregman distance of a strictly convex non-negative differentiable functional [6]. This holds in particular when the loss function is given by the squared $L^2$-norm. Next, another common choice is the MAP estimator that maximizes the posterior, so it corresponds to the most likely reconstruction given the data. On the other hand, a maximum likelihood estimator maximizes the negative log-likelihood of data, i.e., it corresponds to the model parameter that generates the most likely data. This is an unsuitable estimator in ill-posed inverse problems since it frequently leads to overfitting.
To summarize, we will henceforth consider a reconstruction method that minimizes Bayes risk and, as already mentioned, this equals the conditional mean when using a $L^2$-loss.

5.3 Challenges with Bayesian inversion

In the Bayesian setting (section 5.1), both the true model parameter and data are assumed to be generated by random variables, and the goal is to recover the conditional probability of the model parameter given data (posterior) [48, 25, 88, 18, 13]. In contrast, classical (deterministic) approaches view an inverse problem as an operator equation of the type (1) [23, 49, 85, 52] where data may be generated by a random variable, but there are no statistical assumptions on model parameters.

The Bayesian viewpoint offers a more complete analysis than the classical approach that is based on solving (1) in the sense that the posterior describes all possible solutions. In particular, different reconstructions can be obtained by using different estimators and there is a natural framework for uncertainty quantification, e.g., by computing Bayesian credible sets. Furthermore, small changes in the data lead to small changes in the posterior distribution in a fairly general setting [18, Theorem 16] (continuity of the posterior distribution in the Hellinger metric), so working with probability measures on the model parameter space (posterior) and adopting a suitable prior stabilizes an ill-posed inverse problem.

The posterior is, on the other hand, often quite complicated with no closed form expression. Much of the contemporary research therefore focuses on realizing the above advantages with Bayesian inversion without having access to the full posterior. Key topics are designing a “good” prior $\pi_0 \in \mathcal{P}_X$ and to have computationally feasible means for exploring the posterior.

5.3.1 Designing good priors

The difficulty in selecting an appropriate prior lies in capturing the relevant a priori information. Many of the results from the statistical community focus on characterizing priors that lead to Bayesian inference methods with desirable asymptotic properties, like consistency and good contraction rates.

Bayesian non-parametric theory provides a large class of handcrafted priors, see, e.g., [30, chapter 2], [18, section 2], and [48, 13]. These however only capture a fraction of the a priori information that is available. To illustrate this claim, a natural a priori information in medical imaging is that the object being imaged is a human being. It is very difficult, if not impossible, to explicitly construct a prior that encodes this information.

An alternative approach is to consider a prior that is learned from examples in $X$ through some predictive generative model. A simplistic way is to select a Gaussian density that matches the first two sample moments [12]. More elaborate approaches can be based on generative adversarial networks that are trained on unsupervised
data, e.g., a generative adversarial network can be used to learn a Gibbs type of prior in a MAP estimator [63].

### 5.3.2 Computational feasibility

Exploring the posterior requires sampling from a high dimensional probability distribution. It is not possible to directly simulate from the posterior distribution in the infinite dimensional setting unless the model parameter is decomposed into more elementary finite-dimensional components. This quickly becomes computationally challenging in large scale problems, like in imaging where the posterior is a probability distribution over the set of images.

Computational methods used for Bayesian inversion often combine analytic approximations of the posterior with various Markov chain Monte Carlo (MCMC) techniques, see [18, section 5] for a nice survey. There is an extensive theory that guarantees that these techniques are statistically consistent, but it comes with two critical drawbacks that has prevented widespread usage of MCMC techniques in imaging. First, many approaches require access to the prior in closed form, and as already argued for (section 5.3.1), such handcrafted priors are woefully inadequate in representing natural images. Second, these methods are still not sufficiently scalable for exploring the posterior in an efficient manner in large scale inverse problems, such as those that arise in 2D/3D tomographic imaging [8, chapter 1]. Alternatively, one can approximate the posterior with more tractable distributions (deterministic inference), which includes variational Bayes [28] and expectation propagation [69]. Variational Bayes methods have in particular emerged as a popular alternative to the classical MCMC methods, see [9] for some guidance (on p. 860) on when to use MCMC or variational Bayes.

To summarise, one can sometimes with reasonable efficiency compute point estimators that do not involve any integration over the model parameter space, like a MAP estimator. Estimators requiring such integration, like the estimator that minimize Bayes risk, are however computationally unfeasible. This also includes computational steps relevant for uncertainty quantification.

### 5.4 Learned iterative methods

As outlined in section 5.3, there are two challenges associated with using Bayesian inversion: selecting a “good” prior (section 5.3.1) and providing a computationally feasible approach for computing suitable estimators, like the one that minimizes Bayes risk (section 5.3.2).

As we outline here, learned iterative methods address both these challenges. It makes use of techniques from machine learning, and deep neural networks in particular, which have demonstrated a remarkable capacity in capturing intricate relations from example data [57]. A key element is usage of highly parametrized generic models that can be adapted to specific decision rules, such as reconstruction by (7), by
training against example data. Learned iterative methods use a deep neural network to define an estimator (reconstruction method) that minimizes Bayes risk while accounting for the knowledge about how data is generated.

To give a more precise description, consider the joint law \( \mu = \pi_0 \otimes \mathcal{M}(x) \) in (7) used for defining Bayes risk. In most practical applications, this joint law is unknown. Often one may however have access to the corresponding empirical measure given by supervised training data \((x_1, y_1), \ldots, (x_m, y_m) \in X \times Y \) generated by \((x, y) \sim \mu\). This avoids introducing a handcrafted prior \( \pi_0 \in \mathcal{P}_X \). Furthermore, searching over all non-randomized decision rules is computationally unfeasible. Instead, we restrict our attention to those given by a (deep) neural network architecture, which are known to have large capacity (can approximate any Borel measurable mapping arbitrarily well [75]) and there are computationally feasible implementations. To summarize, we have a family of reconstruction methods \( \mathcal{A}_{\theta}^*: Y \to X \) parametrized by a finite dimensional parameter set \( \Theta \) and the optimal one is given by solving the training problem

\[
\theta^* \in \arg \min_{\theta \in \Theta} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell_X(x_i, \mathcal{A}_{\theta}(y_i)) \right\}.
\]  

The above approach for defining a reconstruction operator \( \mathcal{A}_{\theta}^*: Y \to X \) is fully data driven in the sense that neither a prior on model parameter space nor a data model are handcrafted beforehand. Instead, all information is derived from the training data, which in particular does not utilize knowledge about how data is generated. This becomes a serious issue when the number of independent samples in training data are low compared to number of unknowns, which is commonly the case in imaging. Next, in many inverse problem the data model \( x \mapsto \mathcal{M}(x) \) that describes how data is generated is known. Thus, it is unnecessarily pessimistic to disregard this information as in a fully data driven approach to reconstruction.

Learned iterative schemes [1, 2] define a non-linear reconstruction operator parametrized by a deep convolutional neural network architecture that accounts for the data model, or more precisely the data likelihood. The idea is to unroll a fixed point iterative scheme relevant for solving the inverse problem and replace the explicit iterative updating rule with a learned one given by a deep convolutional residual network. The approach can be formulated as a general scheme for solving (possibly non-linear) inverse problems [1, 2, 35], see also [65, 66, 20] for a formulation that learns proximal updates in linear inverse problems. This results in a computationally feasible approach with surprisingly low requirements on training data and good generalization properties that outperforms state-of-the-art image reconstruction in CT [1, 2, 35], MRI [64, 65, 37, 66], photoacoustic tomography [38], and superresolution [64, 65, 20].
6 Tasks on model parameters

We consider tasks formulated as an operator that acts on model parameter space $X$ and that takes values in a set $D$ (decision space). We will start with the abstract formalization of such tasks using the language of statistical decision theory. Similar to how reconstruction was treated (section 5.2), the task is represented by a non-randomized decision rule and we will select the one that minimizes Bayes risk. Next, we also indicate how such decision rules can be computed efficiently using (deep) neural networks and supervised learning that minimizes the empirical risk. The remainder of the section is devoted to providing examples that concretizes the abstract framework and illustrates its general applicability.

6.1 Abstract setting

Let $(\{(X, \mathcal{S}_X), \{P_z\}_{z \in \Delta}\})$ be a statistical model where the model parameter space $(X, \mathcal{S}_X)$ is a measurable space and $\{P_z\}_{z \in \Delta} \subset \mathcal{P}_X$ is some family of probability measures on $X$ parametrized by elements in $\Delta$. Next, there is a measurable space $(D, \mathcal{S}_D)$ (decision space) and a fixed (task adapted) loss function ([60, Definition 3.2])

$$L_D: \Delta \times D \to \mathbb{R} \text{ where } L_D(z, d) := \ell_D(\tau(z), d) \quad (9)$$

with given $\tau: \Delta \to D$ and $\ell_D: D \times D \to \mathbb{R}$ (decision distance). The statistical model along with the decision space and loss function defines a statistical estimation problem. Many tasks can now be seen as an appropriate non-randomized decision rule $T: X \to D$ (task operator).

Before proceeding, it is worth reflecting over the roles of the above sets. In our set-up, the decision making associated with the task is based on elements in the decision space $D$ whereas actual observables are elements in $X$, so the task is represented by a measurable mapping $T: X \to D$ (task operator). Often it is more natural to formalize the task as a mapping $\tau: \Delta \to D$ where elements in the set $\Delta$ are related to those in $X$. A difficulty is that elements in $\Delta$ are not observable and the mapping relating its elements to those in $X$ is unknown. Hence, the challenge is to infer an appropriate mapping $T$ given $\tau$ by resorting to some suitable “optimality” principle. The examples in section 6.2 will further clarify the various roles of these sets in decision making.

Just as in section 5.2, we consider a decision rule that minimizes Bayes risk. More precisely, assume $\Delta$ is itself a measurable space and consider a fixed probability measure $\eta_0 \in \mathcal{P}_\Delta$ (task prior). The task operator is the non-randomized decision rule $T: X \to D$ that minimizes the associated Bayes risk:

$$R_{\eta_0}(T) := \mathbb{E}_{\eta_0 \otimes P_z} \left[ \ell_D(\tau(z), T(x)) \right] \quad \text{where } (z, x) \sim \eta_0 \otimes P_z. \quad (10)$$

A difficulty is to provide a ‘reasonable’ task prior $\eta_0 \in \mathcal{P}_\Delta$. Another is that $P_z \in \mathcal{P}_X$ is not known. Hence, one needs to consider the joint law $\eta := \eta_0 \otimes P_z$ in (10) as an
unknown. Note that this differs from reconstruction, where the joint law is either known (as in section 5.1), or the prior is unknown but the data likelihood is known (as in section 5.4). Since the joint law is unknown, we replace it by the empirical measure given by (supervised) training data \((z_1, x_1), \ldots, (z_m, x_m) \in \triangle \times X\), i.e., one has i.i.d. samples generated by a \((\triangle \times X)\)-valued random variable \((z, x) \sim \eta\). Furthermore, due to issues associated with computational feasibility (section 5.3.2), we consider a parametrized family of decision rules \(T_\vartheta : X \to D\) given by a (deep) neural network architecture. Then, the task operator is the decision rule \(T_{\vartheta^*} : X \to D\) parametrized by a finite dimensional parameter in \(\Xi\) and the optimal one \(\vartheta^* \in \Xi\) is given by empirical risk minimization:

\[
\vartheta^* \in \arg \min_{\vartheta \in \Xi} \left\{ \frac{1}{m} \sum_{i=1}^m \ell_D (\tau(z_i), T_\vartheta(x_i)) \right\}. \tag{11}
\]

We conclude with examples showing how a wide range of image processing tasks can be phrased as decision rules in a statistical decision problem.

### 6.2 Examples

The abstract framework in section 6.1 for formalizing a task on model parameter space is very generic and covers a wide range of possible tasks. In the following, we list concrete examples from imaging in order to show how this framework can be adapted to specific cases. To ensure a computational feasible implementation, our focus is on tasks that have been successfully addressed using techniques from deep learning. Deep learning has proven to be an efficient computational framework for many tasks, much thanks to its ability to progressively learn discriminative hierarchical features of the input data by means of training a suitable deep neural network. Hence, this limitation is not as restrictive as it may seem at a first glance, which will also become evident by the examples listed here and in section 6.3.

Unless otherwise stated, tasks are formulated for grey-scale images defined on a fixed domain \(\Omega \subset \mathbb{R}^n\), i.e., \(X := L^2(\Omega, \mathbb{R})\). We will also assume that \(X\) is a measurable space for some \(\sigma\)-algebra \(\mathcal{S}_X\). Finally, \(\mathcal{M}\) denotes the space of measurable mappings, e.g., \(\mathcal{M}(X, D)\) is \(D\)-valued measurable mappings defined on \(X\).

#### 6.2.1 Classification

The task is to classify an image into one of \(k\) distinct labels, or more precisely, associate an image to a probability distribution over all \(k\) labels. This task is represented by a non-randomized decision rule in a statistical estimation problem where \(\Delta := \mathbb{Z}_k\) and the decision space \(D := \mathcal{P}_\Delta\) is probability distributions over the \(k\) labels. The task adapted loss function is given by (9) with

\[
\ell_D(d, d') := - \sum_{z \in \Delta} d(z) \log d'(z) \text{ for } d, d' \in D \quad \text{and} \quad \tau(z) := \delta_z \text{ for } z \in \Delta.
\]
Bayes risk in (10) associated with a decision rule \( T: X \rightarrow D \) for given task prior \( \eta_0 \in \mathcal{P}_\Delta \) becomes

\[
R_{\eta_0}(T) := E_{\eta_0 \otimes P_z} \left[ \ell_D(\tau(z), T(x)) \right] = \int_X \int_\Delta \left[ -\log [T(x)(z)] \right] d\eta_0(z) dP_z(x).
\]

The corresponding empirical risk minimization in (11) is

\[
\theta^* \in \arg \min_{\theta \in \Xi} \left\{ \frac{1}{m} \sum_{i=1}^m \left[ -\log [T(x_i)(z_i)] \right] \right\} \text{ for training data } (z_i, x_i) \in \Delta \times X. \quad (12)
\]

There are several papers dealing with how to construct a suitable (deep) neural network architecture for the set of decision rules \( D = \{T_\theta\}_{\theta \in \mathbb{R}^N} \) and solving (12) will then correspond to training a classifier, see [58] for an early approach based on a convolutional neural network, AlexNet [54] and ResNet [39] represent examples of further development along this line.

### 6.2.2 Semantic segmentation

The task here is to classify each point in an image into one of \( k \) possible labels, so the special case \( k = 2 \) corresponds to (binary) segmentation. Stated more formally, semantic segmentation applies a mapping that associates each point in an image in \( X \) to a probability distribution over all \( k \) labels. This task becomes a non-randomized decision rule in a statistical estimation problem where \( \Delta := \mathcal{M}(\Omega, \mathbb{Z}_k) \) and the decision space \( D := \mathcal{M}(\Omega, \mathcal{P}_{\mathbb{Z}_k}) \) is the set of measurable mappings from \( \Omega \) to the class of probability measures on \( \mathbb{Z}_k \). The task adapted loss function is given by (9) with

\[
\ell_D(d, d') := \int_\mathbb{Z}_k \left[ -\sum_{i \in \mathbb{Z}_k} d(t)(i) \log[d'(t)(i)] \right] dt \text{ for } d, d': \Omega \rightarrow \mathcal{P}_{\mathbb{Z}_k},
\]

\[
\tau(z)(t) := \delta_{z(t)} \text{ for } z: \Omega \rightarrow \mathbb{Z}_k \text{ and } t \in \Omega.
\]

The decision distance \( \ell_D: D \times D \rightarrow \mathbb{R} \) simply integrates the point-wise cross entropy of the (point-wise) independent probability measures \( d(t) \) and \( d'(t) \). The cross entropy is a well-known notion from information theory for quantifying the dissimilarity between probability distributions [16] and it is often used as a learning objective in generative models involving probability distributions.

Bayes risk in (10) associated with a decision rule \( T: X \rightarrow D \) for a given task prior \( \eta_0 \in \mathcal{P}_\Delta \) can then be written as

\[
R_{\eta_0}(T) := E_{\eta_0 \otimes P_z} \left[ \ell_D(\tau(z), T(x)) \right] = \int_X \left[ \int_\Delta \left[ \int_\Omega -\log [T(x)(z)] \right] dt \right] d\eta_0(z) dP_z(x).
\]
The corresponding empirical risk minimization in (11) is

$$\vartheta^* \in \arg \min_{\vartheta \in \Xi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \int_{\Omega} -\log \left[ T_\vartheta(x_i)(z_i(t)) \right] dt \right\}. \quad (13)$$

Note that $T(x)(t)$ is a probability distribution over $Z_k$ and $z(t) \in Z_k$ when $z \in \Delta$, so in particular $T(x)(t)(z(t)) \in [0, 1]$ for any $t \in \Omega$.

The set of decision rules $\mathcal{D} = \{ T_\vartheta \}$ can be parametrized by (deep) neural networks, in which case solving (13) corresponds to training a segmentation operator. Deep neural net architectures suitable for semantic segmentation are presented in [61, 74, 84], see also the surveys in [91, 34]. In particular, the SegNet architecture has been successful for semantic segmentation of 2D images [5]. For (binary) segmentation one may use the U-net [81, 15].

### 6.2.3 Anomaly detection

The task here is to detect the difference (anomaly) between two grey-scale images, so $X = L^2(\Omega, \mathbb{R}) \times L^2(\Omega, \mathbb{R})$ for a fixed domain $\Omega \subset \mathbb{R}^n$. This becomes a non-randomized decision rule in a statistical estimation problem where $\Delta := X$ and the anomaly is represented by grey-scale images, so the decision space is $D := L^2(\Omega, \mathbb{R})$.

The task adapted loss function is given by (9) with

$$\ell_D(d, d') := \| d - d' \|_\Delta^2$$

for $d, d' \in D$ and $\tau(z) := z_1 - z_2$ for $z = (z_1, z_2) \in \Delta$.

Bayes risk in (10) associated with a decision rule $T: X \to D$ for given task prior $\eta_0 \in \mathcal{P}_\Delta$ becomes

$$\mathcal{R}_{\eta_0}(T) := \mathbb{E}_{\eta_0 \otimes \mathbb{P}_z} \left[ \ell_D(\tau(z), T(x)) \right]$$

$$= \int_X \int_\Delta \left\| (z_1, z_2) - T(x_1, x_2) \right\|_D^2 \mathbb{d}P_z(z) \mathbb{d}P_x(x_1, x_2)$$

and note that $z = (z_1, z_2) \in \Delta$ and $x = (x_1, x_2) \in X$. The corresponding empirical risk minimization in (11) is

$$\vartheta^* \in \arg \min_{\vartheta \in \mathbb{R}^N} \left\{ \frac{1}{m} \sum_{i=1}^{m} \left\| (x_1 - x_2) - T_\vartheta(x_1, x_2) \right\|_D^2 \right\}. \quad (14)$$

where $(x_1^i, x_2^i) \in X$ is the supervised training data. One may here use a (deep) convolutional neural net to parametrize the decision rules in $\mathcal{D} = \{ T_\vartheta \}_{\vartheta \in \mathbb{R}^N}$.

### 6.2.4 Caption generation

Caption generation refers to the task of associating an image to an appropriate sentence, or paragraph, that describes its content. More precisely, define $W$ to be
the set of words in a chosen language, enlarged by a “stop word” \( w_{\text{stop}} \) that marks the end of the caption. Next, let \( \mathcal{C} \) denote the set of captions, which are finite sequences made up of elements from \( \mathcal{W} \) where each sequence contains \( w_{\text{stop}} \) exactly once as its last element. Then, caption generation is the task of mapping an image to a sequence in \( \mathcal{C} \).

This task becomes a non-randomized decision rule in a statistical estimation problem where \( \Delta := \mathcal{C} \) is the set of captions, so \( \mathbb{P}_z \) is the probability of a caption \( z \). The decision space is \( D := \mathcal{P}_\mathcal{C} \), i.e., probability distributions over set of captions \( \mathcal{C} \), and the task adapted loss function is given by (9) with

\[
\ell_D(d, d') := - \sum_{c \in \mathcal{C}} d(c) \log d'(c) \quad \text{for } d, d' \in D,
\]

\[
\tau(z) := \delta_z \in \mathcal{P}_\mathcal{C} \quad \text{for a caption } z \in \Delta.
\]

To express Bayes risk that is to be minimized when computing the optimal decision rule, consider an element \( z \in \Delta = \mathcal{C} \) (caption) and let \( z_i \in \mathcal{W} \) denote its \( i \)-th word. Next, let \( \mathcal{W}_n \subset \mathcal{W} \) denote the set of sequences of \( n \) words that do not contain the stop word \( w_{\text{stop}} \) (unfinished sentences), i.e.,

\[
\mathcal{W}_n := \{(w_i)_{i=1}^n \subset \mathcal{W} \mid w_i \neq w_{\text{stop}} \text{ for } i = 1, \ldots, n\}.
\]

Since an element in the decision space \( d \in D := \mathcal{P}_\Delta \) is a probability measure on sequences of words (captions), it will in particular yield the following probability measure on \( \mathcal{W}_n \):

\[
d_n((w_i)_{i=1}^n) := d(\{z \in \Delta \mid z_i = w_i \text{ for } i = 1, \ldots, n\}) \quad \text{for } (w_i)_{i=1}^n \in \mathcal{W}_n.
\]

We now consider the measure for \( n + 1 \) sequences that are conditioned on its first \( n \) terms, i.e., let \( (w_i)_{i=1}^n \in \mathcal{W}_n \) be fixed with \( d_n((w_i)_{i=1}^n) > 0 \). Then, \( d \) induces to a probability measure \( \pi_d \in \mathcal{P}_\mathcal{W} \) on the set of words \( \mathcal{W} \) via

\[
\pi_d(w \mid (w_i)_{i=1}^n) := \frac{d_{n+1}(w_1, \ldots, w_n, w)}{d_n(w_1, \ldots, w_n)} \quad \text{for } w \in \mathcal{W} \text{ and } (w_i)_{i=1}^n \in \mathcal{W}_n.
\]

With this notation, one can identify an element \( d \in \mathcal{P}_\Delta \) with its corresponding representation in \( \mathcal{X}_n \mathcal{P}_\mathcal{W}(\cdot \mid \mathcal{W}_n) \) by the identity

\[
d(z) = \prod_i \pi_d(z \mid (z_1, \ldots, z_{i-1})).
\]

Here \( \mathcal{P}_\mathcal{W}(\cdot \mid \mathcal{W}_n) \) is the set of probability measures on \( \mathcal{W} \) conditioned on \( \mathcal{W}_n \).

Bayes risk in (10) associated with a decision rule \( T \colon X \to D \) (potential caption generation operator) for given task prior \( \gamma_0 \in \mathcal{P}_\Delta \) can now be expressed through
conditional densities on \( W \):

\[
R_{\eta_0}(T) := \int_{\triangle} \int_X \ell_D(\tau(z), T(x)) dP_x(x) d\eta_0(z)
\]

\[
= \int_{\triangle} \int_X -\log T(x) dP_x(x) d\eta_0(z)
\]

\[
= \int_{\triangle} \int_X -\log \prod_i \pi_{T(x)}(z \mid z_1, \ldots, z_{i-1}) dP_x(x) d\eta_0(z)
\]

\[
= \int_{\triangle} \int_X -\sum_i \log \pi_{T(x)}(z \mid z_1, \ldots, z_{i-1}) dP_x(x) d\eta_0(z).
\]

To derive the corresponding empirical risk minimization problem, we consider a fixed class of decision rules that are given via a parametrization of their representation in terms of marginal densities. More precisely, given a parameter set \( \vartheta \in \mathbb{R}^m \), the decision rule \( T_\vartheta \) is given by

\[
T_\vartheta(x)(z) = \prod_i \Psi_\vartheta(x; z_1, \ldots, z_{i-1})
\]

where \( x \in X \) and \( z \in \triangle \), and \( \Psi_\vartheta \) is parametrized by a recurrent neural network, for example using the long short-term memory (LSTM) architecture [42]. The corresponding empirical risk minimization in (11) is now given by

\[
\vartheta^* \in \mathop{\arg\min}_{\vartheta \in \mathbb{R}^m} \left\{ \int_{\triangle} \int_X -\sum_i \log \pi_{T_\vartheta(x)}(z_1, \ldots, z_{i-1}) dP_x(x) d\eta_0(z) \right\}. \tag{15}
\]

Solving (15) corresponds to training a image caption generator [92], see also [51].

### 6.3 Further imaging tasks

A common trait with the examples worked out in section 6.2 is that each of them can be recast as finding an optimal decision rule in a statistical estimation problem. Furthermore, deep neural networks offer a computationally feasible implementation of estimating this decision rule from supervised training data.

There is a wide range of other tasks beyond those mentioned in section 6.2 that can be represented as a non-randomized decision rule, which in turn is efficiently parametrized by a suitable deep neural network architecture. The (incomplete) list below is based on [4, 77] and aims to show the diversity of tasks from computer vision that can be approached successfully using a suitable deep neural network architecture.

**Inpainting:** This is essentially interpolation/extrapolation to recover lost or deteriorated parts of images and videos and approaches based on trainable neural networks [97].
Depixelization/super-resolution: The task is here to upsample, i.e., to synthesize realistic details into images while enhancing their resolution [17] or to fill out information “between” pixels by increasing the resolution of the final picture, also known as the single image super-resolution problem [79].

Demosaicing: The task here is to reconstructing a full color image from the incomplete color samples output from an image sensor [90]. Almost all digital cameras, ranging from smartphone cameras to the top-of-the-line digital SLR cameras, use a demosaicing algorithm to convert the captured sensor information into a color image.

Colourising: The task is to apply color to grey scale photos and videos [45].

Image translation: The task is to translate between two classes of images of the same object, e.g., CT and MRI images [95].

Object recognition: This visual classification task involves localization, detection and classification and this can be seen as an example of constellation models, which are a general class of model that describe objects in terms of a set of parts and their spatial relations [77, section 20.5]. An integrated framework based on deep convolutional networks for detection and localizatio was already introduced in [86], see also [40] for recognition and [26] for scene understanding. The most well-known use case is recognition of multiple faces in an image where statistical shape models play a central role [100, 22, 94] An analogous task relevant for clinical image guided diagnostics is detecting melanoma from images of skin lesions [24, 36].

Non-rigid image registration: The task here is to deform a template image in a “natural” way so that it matches a target image. This is a key step in spatiotemporal imaging and deep neural networks have been utilized for this purpose [29, 98].

Parametric regression: The task here is to statistically determine relationships among variables (parameters) in a statistical model. This is important in clinical diagnostics where one seeks to determine risk factors and biomarkers of diagnostic and prognostic value associated with clinical progression and severity of specific diseases. An example of image guided regression is estimating cardiovascular risk factors, such as age, from retinal fundus photographs [76]. Another is predicting patient overall survival as in the 2018 Multimodal Brain Tumor Segmentation Challenge based on BRATS imaging data [68], and predicting scores for Alzheimer’s disease from imaging data [67, 59].

The abstract framework in section 7.1 allows one to perform any of the above tasks (and those in section 6.2) jointly with a reconstruction step for solving an inverse problem. Examples of the latter are deconvolution, Fourier inversion (MRI imaging), or more elaborate schemes for various types of tomographic imaging. A key success factor is access to suitable training data, another is usage of a differentiable loss function along with a trainable differentiable parametrization (deep neural network architecture) of the task operator.
7 Task adapted reconstruction

As stated in the introduction (section 1), solving an inverse problem (reconstruction) is one step in a pipeline (fig. 1) that often involves further coupled tasks necessary for decision making. Task adapted reconstruction refers to methods that integrate the reconstruction with (parts of) the decision making procedure, thereby adapting the reconstruction method for the specific task at hand.

7.1 Abstract setting

We will present a generic framework for task adapted reconstruction that is computationally feasible and adaptable to specific inverse problems and tasks. A key element is to formalize both the reconstruction and task as non-randomized decision rules within a statistical estimation problem.

More precisely, our starting point is the inverse problem in section 5 where the data model in (3) is known. Following section 5.2, the reconstruction step can be seen as a decision rule in a statistical estimation problem defined by the statistical model $((Y, S_Y), \{M(x)\}_{x \in X})$, decision space $(X, S_X)$, and loss $\ell_X : X \times X \to \mathbb{R}$. Given a prior $\pi_0 \in \mathcal{P}_X$ allows us to define a reconstruction method as a non-randomized decision rule that minimizes the $\pi_0$-average risk (Bayes risk), i.e., as a mapping that solves

$$\hat{A} \in \arg \min_{A \in \mathcal{A}(Y,X)} \left\{ E_{\pi_0 \otimes M(x)} \left[ \ell_X(x, A(y)) \right] \right\} \text{ where } (x,y) \sim \pi_0 \otimes M(x). \tag{16}$$

Likewise, following section 6.1 a task becomes a decision rule in a statistical estimation problem defined by the statistical model $((X, S_X), \{P_z\}_{z \in \Delta})$, decision space $(D, S_D)$, and loss given by (9) with known $\tau : \Delta \to D$ (feature extraction map) and $\ell_D : D \times D \to \mathbb{R}$ (decision distance). Given a task prior $\eta_0 \in \mathcal{P}_\Delta$, the non-randomized decision rule representing the task (task operator) can be seen as a minimizer to the $\eta_0$-average risk (Bayes risk):

$$\hat{T} \in \arg \min_{T \in \mathcal{T}(X,D)} \left\{ E_{\eta_0 \otimes P_x} \left[ \ell_D(\tau(z), T(x)) \right] \right\} \text{ where } (z,x) \sim \eta_0 \otimes P_x. \tag{17}$$

We now have the following three approaches to task adapted reconstruction.

Sequential approach: A sequential approach starts with determining the reconstruction operator, and then uses it to define the task operator. It is based on assuming that statistical assumptions for $(z,x) \sim \eta_0 \otimes P_z$ and $(x,y) \sim \pi_0 \otimes M(x)$ are consistent, e.g., by assuming that $P_z$ is the push forward of $M(x)$ through the reconstruction operator\(^2\). The task adapted reconstruction is then given as

$$\hat{T} \circ \hat{A} : Y \to D \tag{18}$$

\(^2\) One may here consider alternate assumptions, like assuming that $\pi_0 \in \mathcal{P}_X$ can be obtained by marginalizing the measure $\eta_0 \otimes P_z$ over $\Delta$ using $\eta_0 \in \mathcal{P}_\Delta$.  

where \( \hat{A}^\dagger \in \mathcal{A}(Y, X) \) solves (16) and \( \hat{T} \in \mathcal{A}(X, D) \) solves (17).

**End-to-end approach:** The fully end-to-end approach ignores the distinction between reconstruction and the task. Assuming \((z, y) \sim \nu\) for some measure \(\nu \in \mathcal{P}_{\Delta \times Y}\), the task adapted reconstruction is here given as \(\hat{B} : Y \to D\) that solves

\[
\hat{B} \in \arg \min_{B \in \mathcal{A}(Y, D)} \left\{ E_{\eta_0 \otimes P_z} \left[ \ell_D (\tau(z), B(y)) \right] \right\}
\]  

where \((z, x) \sim \nu\). (19)

**Joint approach:** The joint approach is a middle-way between the sequential and end-to-end approaches. It is based on assuming that there is a joint law \((z, x, y) \sim \sigma\), which by the chain rule in probability can be written in terms of conditional probabilities:

\[
d\sigma(z, x, y) = d\pi(y \mid z, x) d\pi(x \mid z) d\pi(z).
\]

Assume that \(x\) is a sufficient statistic for \(y\), i.e., \(d\pi(y \mid z, x) = d\pi(y \mid x)\). Combined with \((z, x) \sim \eta_0 \otimes P_z\) and \((x, y) \sim \pi_0 \otimes \mathcal{M}(x)\), we obtain

\[
d\sigma(z, x, y) = d\mathcal{M}(x)(y) dP_z(x) d\eta_0(z).
\]

Next, we introduce a **joint loss** that interpolates between the sequential case and the end-to-end case. Specifically, we let \(\ell_{\text{joint}} : (X \times D) \times (X \times D) \to \mathbb{R}\) be given as

\[
\ell_{\text{joint}} \left( (x, d), (x', d') \right) := (1 - C)\ell_X (x, x') + C\ell_D (d, d') \quad \text{for fixed } C \in [0, 1].
\]  

(20)

Task adapted reconstruction is now given by (18) where the operators jointly solve

\[
\left( \hat{A}^\dagger, \hat{T} \right) \in \arg \min_{T \in \mathcal{A}(X, D)} \mathbb{E}_x \left[ \ell_{\text{joint}} \left( (x, \tau(z)), (A^\dagger(y), T \circ A^\dagger(y)) \right) \right].
\]  

(21)

Note first that in the limit \(C \to 0\), the joint approach becomes the sequential one. Next, it may seem sufficient to only consider the loss \(\ell_D\) in (21), i.e., to set \(C = 1\) in (20), which recovers the end-to-end approach. There is however a problem with non-uniqueness in this case since

\[
\left( \hat{A}^\dagger, \hat{T} \right) \text{ solves (21)} \implies \left( B^{-1} \circ \hat{A}^\dagger, \hat{T} \circ B \right) \text{ solves (21)} \text{ for any invertible } B : X \to X.
\]  

(22)

This non-uniqueness does not arise when \(C < 1\), so incorporating a loss term associated with the reconstruction may act as a regularizer. This also indicates that the limit \(C \to 1\) does not necessarily coincide with the case \(C = 1\).
7.2 Computational implementation

In section 5.3.1 we mentioned the difficulty to select an appropriate prior \( \pi_0 \in \mathcal{P}_X \) for Bayesian inversion whereas the measure \( M(x) \in \mathcal{P}_Y \) is often known by the data model. Furthermore, both measures \( \eta_0 \in \mathcal{P}_\Delta \) and \( \mathbb{P}_z \in \mathcal{P}_X \) must be considered as unknown for most tasks. Hence any realistic scenario would contain \( \sigma \) as an unknown. An option is to replace these measures by their empirical counterparts given by suitable supervised training data.

Another concern is computational feasibility. The optimizations in (16), (17), (19) and (21) are taken over all measurable mappings between relevant spaces, which is computationally unfeasible. This can be addressed by considering parametrized sets of measurable mappings as done in sections 5.4 and 6.1. More precisely, we employ a learned iterative scheme to parametrize a family of reconstruction methods \( A^\dagger_{\theta} : Y \rightarrow X \) since this parametrization includes knowledge about the data model (section 5.4). Likewise, decision rules associated with the task are given by an appropriate parametrized family of mappings \( T_{\varphi} : X \rightarrow \mathcal{D} \). Finally, the approach for the end-to-end setting is to directly parametrize \( \mathcal{B}_{\varphi} : Y \rightarrow \mathcal{D} \).

Using such parametrizations allows one to reformulate (16) and (17) as (25), (19) as (25), and (21) as (27). A key aspect for the implementation is to use stochastic gradient descent (SGD) based methods for finding appropriate parameters by approximately solving the empirical versions of (19), (25) and (27). This requires that the above parametrizations are differentiable, which in particular requires using differentiable loss-functions.

Depending on the type of supervised training data, we can now pursue either of the three approaches listed in section 7.1.

**Sequential approach:** Here we have access to two sets of supervised training data that are coupled:

\[
(x_i, y_i) \in X \times Y \quad \text{generated by } (x, y) \sim \pi_0 \otimes M(x) \text{ for } i = 1, \ldots, m
\]

\[
(z_i, x_i) \in \Delta \times X \quad \text{generated by } (z, x) \sim \eta_0 \otimes \mathbb{P}_z \text{ for } i = 1, \ldots, m.
\]

The coupling is that \( x_i \)'s in the second data set (bottom one) are reconstructions obtained from \( y_i \)'s in the first data set (top one). This ensures consistency with the statistical assumptions mentioned before for the sequential approach. The reconstruction is then given by the mapping

\[
T_{\varphi^*} \circ A^\dagger_{\theta^*} : Y \rightarrow \mathcal{D}
\]

where \( \theta^* \in \Theta \) solves (8) and \( \varphi^* \in \Xi \) solves (11), i.e.,

\[
\theta^* \in \arg \min_{\theta \in \Theta} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell_X(x_i, A^\dagger_{\theta}(y_i)) \right\}
\]

\[
\varphi^* \in \arg \min_{\varphi \in \Xi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell_D(\tau(z_i), T_{\varphi}(x_i)) \right\}.
\]
Note here that the only requirement for the sequential approach is that the assumptions \((z, x) \sim \eta_0 \otimes P_z\) and \((x, y) \sim \tau_0 \otimes \mathcal{M}(x)\) are jointly consistent. The learned task operator given by solving for \(\vartheta^*\) in (25) is only well defined for input taken from the support of it’s training data, so it may fail when applied to data it has never seen. This is especially the case if new data has a different statistical assumption. Hence, it is important to ensure the range of the reconstruction operator is contained in the support of the elements \(x \in X\) used to train the task. In most practical implementations, this is ensured by simply letting \(x_i\)’s in \((z_i, x_i)\) in (23) be the output of the learned reconstruction operator \(A_{y^*}^\dagger : X \rightarrow Y\).

**End-to-end approach:** Supervised training data is here of the form
\[(z_i, y_i) \in \triangle \times X \text{ generated by } (z, x) \sim \eta_0 \otimes P_z \text{ for } i = 1, \ldots, m. (26)\]

The reconstruction is given by \(B_{\vartheta} : Y \rightarrow D\) where \(\vartheta^* \in \Xi\) solves
\[\vartheta^* \in \arg\min_{\vartheta \in \Xi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell_D(\tau(z_i), B_{\vartheta}(y_i)) \right\}. \tag{27}\]

**Joint approach:** In this approach we assume access to supervised training data that jointly involves the reconstruction and task:
\[(x_i, y_i, z_i) \in X \times Y \times \triangle \text{ generated by } (x, y, z) \sim \sigma \text{ for } i = 1, \ldots, m. (28)\]

Given such data, the corresponding reconstruction method can be defined as in (24) where \((\vartheta^*, \theta^*) \in \Theta \times \Xi\) solves the following joint empirical loss minimization:
\[(\theta^*, \vartheta^*) \in \arg\min_{(\theta, \vartheta) \in \Theta \times \Xi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell_{\text{joint}} \left( (x_i, \tau(z_i)), (A_{y^*}^\dagger(y_i), T_{\vartheta} \circ A_{y^*}^\dagger(y_i)) \right) \right\} \tag{29}\]

where \(\ell_{\text{joint}} : (X \times D) \times (X \times D) \rightarrow \mathbb{R}\) is the joint loss in (20). Note that one may in addition have access to separate sets of training data of the form (23) and (28). In such case, it is possible to first pre-train by solving for (8) and (11) separately, and use the resulting outcomes to initialize an algorithm for solving (29).

### 7.3 Applications

In the following we demonstrate performance of the task adapted reconstruction scheme for (24) that is based on solving (29). All cases involve tomographic reconstruction from 2D parallel beam data and as tasks, we consider MNIST classification and segmentation.
7.3.1 Joint tomographic reconstruction and classification

Task: Recover probabilities that a 2D grey scale MNIST image is a 0, 1, ..., 9 from noisy parallel beam tomographic data (see section 6.2.1).

Data: Elements in $Y$ are real-valued functions representing samples of a Poisson random variable with mean equal to the exponential of the parallel beam ray transform and an intensity corresponding to 60 photons/line. The ray transform is digitized by sampling the angular variable at 5 uniformly sampled points in $[0, \pi]$ with 25 lines/angle.

Model parameter space: Elements in $X$ are functions representing images supported on a fixed rectangular region $\Omega \subset \mathbb{R}^2$, so $X := L^2(\Omega, \mathbb{R})$. These are discretized by sampling on a uniform $28 \times 28$ grid. The loss $\ell_X : X \times X \to \mathbb{R}$ is the squared $L^2$-distance on $X$.

Decision space: $\triangle := \{0, 1, \ldots, 9\}$ is the set of labels and $D$ is probability distributions over $\triangle$ with a loss function $\ell_D : D \times D \to \mathbb{R}$ given by the cross entropy:

$$\ell_D(d, d') := -\sum_{i \in \triangle} d(i) \log[d'(i)]$$

for $d, d' \in \mathcal{P}_{\triangle}$.

In addition to cross entropy, we employ classification accuracy to measure performance. Given a probability distribution $d \in D$ over $\triangle$, the single label prediction is defined to be the element in $\triangle$ that is assigned the highest probability, i.e. $\arg \max_{z \in \triangle} d(z)$. The percentage of images in the evaluation data set for which the predicted label coincides with the real one is reported as classification accuracy.

Reconstruction and task operators: Reconstruction $A^\dagger_{\theta} : Y \to X$ is given by the learned gradient descent in [1] and task $T_\vartheta : X \to D$ is a MNIST classifier given by a standard convolutional neural net classifier with three convolutional layers, each followed by 2 $\times$ 2 max pooling for segmentation. The activation functions used were ReLUs, layers had 32, 64 and 128 channels, respectively. The final layer is dense and transforms the output of the last convolutional layer to a logit layer of size 10, with the last activation function being a softmax.

Joint training: Joint supervised data is given as 512,000 triplets $(x_i, y_i, z_i)$ where $z_i \in \triangle$ is the label corresponding to the MNIST labels. We also performed pre-training for both the reconstruction and task operator (classifier). The reconstruction operator was pre-trained using 256,000 pairs $(x_i, y_i)$ with 8000 steps with a batch size of 64 and the task operator (classifier) was pre-trained until 97.7% accuracy. Note here that we use about 60,000 entries from the MNIST database, so the above supervised data is not statistically independent.

Example outcomes, which are summarized in table 1 and fig. 2, show that a joint approach outperforms a sequential one when considering the classification accuracy.
Besides an improved classification accuracy we also see a significant improvement
regarding interpretability. The reconstructed image part can in the joint setting
actually be used as a benchmark to assess the reconstructed classification probabili-
ties. On the other hand, the sequential approach results in classification probabilities
that deviates from this intuitive observation. We also see that in both cases, the
classification probabilities are unnaturally concentrated on a single label, but this is
a know phenomena also for regular for MNIST classification [33].

<table>
<thead>
<tr>
<th>Approach</th>
<th>Accuracy</th>
<th>$L^2$-loss</th>
<th>Cross entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-training</td>
<td>93.61%</td>
<td>9.0</td>
<td>0.643</td>
</tr>
<tr>
<td>Sequential</td>
<td>96.01%</td>
<td>9.0</td>
<td>0.124</td>
</tr>
<tr>
<td>End-to-end</td>
<td>96.70%</td>
<td>19.7</td>
<td>0.118</td>
</tr>
<tr>
<td>Joint with $C = 0.01$</td>
<td>96.74%</td>
<td>12.8</td>
<td>0.108</td>
</tr>
<tr>
<td>Joint with $C = 0.5$</td>
<td>97.00%</td>
<td>9.2</td>
<td>0.100</td>
</tr>
<tr>
<td>Joint with $C = 0.999$</td>
<td>96.61%</td>
<td>9.0</td>
<td>0.108</td>
</tr>
<tr>
<td>Classification on true images</td>
<td>97.76%</td>
<td></td>
<td>0.088</td>
</tr>
</tbody>
</table>

Tab. 1: In both the pre-training and sequential approaches, the reconstruction
and task operators are trained separately. In the sequential approach,
the task operator is then further trained on the output of the trained
reconstruction operator. In the end-to-end approach, which corre-
sponds to $C = 0$ in (20), the reconstruction operator is pre-trained
with $L^2$-loss. Finally, the joint approach uses the full loss (20). We
see that the classification accuracy (explained in “Decision space” in
section 7.3.1) improves when using a joint approach. In fact, using a
“suitable” $C$ (fig. 2(a)) yields an accuracy of 97.00% that is quite close
to the upper limit of 97.76%, which is the accuracy of the classifier
when trained on true images.
Fig. 2: Joint tomographic reconstruction and classification of MNIST images. Training data is to the left and reconstructed image with classification probabilities are on the right. Data on overall performance on all of the MNIST dataset (accuracy) is given in table 1.
Joint tomographic reconstruction and segmentation

Task: Recover the probability map for segmentation of a grey scale image (see section 6.2.2 with \( k = 2 \)) from noisy parallel beam tomographic data. In this specific example, we consider segmenting the grey matter of CT images of the brain, which is relevant in imaging of neurodegenerative diseases like Alzheimer's disease.

Data: Elements in \( Y \) are real-valued functions on lines representing parallel beam tomographic data, which are digitized by sampling the angular variable at 30 uniformly sampled points in \([0, \pi]\) with 183 lines/angle. We furthermore add 0.1% additive Gaussian noise to data.

Model parameter space: Elements in \( X \) are functions representing images supported on a fixed rectangular region \( \Omega \subset \mathbb{R}^2 \), so \( X := L^2(\Omega, \mathbb{R}) \). These are discretized by uniform sampling on a 128 \( \times \) 128 grid. The loss \( \ell_X : X \times X \rightarrow \mathbb{R} \) is the squared \( L^2 \)-distance.

Decision space: Elements in \( D \) are point-wise probability distributions over binary images on \( \Omega \), which can be represented by grey-scale images with values in \([0, 1]\) that gives the probability that a point is part of the segmented object. Hence, \( D = \mathcal{M}(\Omega, [0, 1]) \) with the loss function \( \ell_D : D \times D \rightarrow \mathbb{R} \) as the cross entropy:

\[
\ell_D(d, d') := \int_\Omega \left[ -\sum_{i=1}^2 d(t)(i) \log [d'(t)(i)] \right] dt \quad \text{for } d, d' \in \mathcal{M}(\Omega, [0, 1]).
\]

Reconstruction and task operators: Reconstruction \( A^\dagger : Y \rightarrow X \) is given by the Learned Primal-Dual scheme in [2] and task \( T_\theta : X \rightarrow D \) is given by an “off the shelf” U-net convolutional neural net for segmentation [81].

Joint training: Joint supervised data is given as 100 triplets \((x_i, y_i, d_i)\) where \( d_i \) is the segmentation (binary image). We extend joint training data by data argumentation (\( \pm 5 \) pixel translation and \( \pm 10^\circ \) rotation). There was no pre-training.

Example outcomes are summarized in figs. 3 and 4. Note that \( C \to 0 \) corresponds to the sequential approach, so the image for \( C = 0.01 \) can be seen as the outcome from a sequential approach. Clearly, a joint approach with \( C \approx 0.5 \) or 0.9 outperforms a sequential one.

Next, as \( C \) decreases the reconstruction becomes more adapted to the task of segmentation. In the limit \( C \to 0 \) the task part is viable but the reconstruction image is useless, which is to be expected. In the other direction, as \( C \) increases the reconstructed image becomes less adapted to the task and the latter becomes increasingly challenging due to the low contrast between white and grey matter.

Finally, using \( C > 0 \) not only reduces the non-uniqueness as explained in (22), it further regularizes in the sense that information from the reconstruction guides
Task adapted reconstruction

Fig. 3: Log-log plot of loss functions for joint reconstruction and segmentation after joint training for different $C$ in (20). Clearly, there is no joint minimizer but $0.5 \lesssim C \lesssim 0.9$ is a good compromise.

the segmentation, which otherwise would amount to learning the segmented image directly from the noisy sinograms. Intuitively there seems to be an “information exchange” between the task of reconstruction and that of segmentation, which when properly balanced by choosing $C$ acts as a regularizer for the segmentation, e.g., the white/grey matter contrast in the reconstruction is overemphasized for small $C$. This improves the interpretability since it shows how the reconstructed image “helps” in interpreting why a certain segmentation is taken.
True image & segmentation.

Data: Sinogram, 30 angles & 177 lines/angle.

Joint reco. & segmentation: $C = 0.01$.

Joint reco. & segmentation: $C = 0.1$.

Joint reco. & segmentation: $C = 0.5$.

Joint reco. & segmentation: $C = 0.9$.

Joint reco. & segmentation: $C = 0.99$.

Joint reco. & segmentation: $C = 0.999$.

Fig. 4: Joint tomographic reconstruction and segmentation for different values of $C$ in (20). The segmentation is a normalized grey-scale image denoting the probability that a point belongs to the segmented structure. The choice $C = 0.9$ seems to be a good compromise for a good reconstruction and segmentation (see fig. 3). Note also that $C \to 1$ gives the sequential approach, so $C = 0.999$ may serve as a proxy for it. Reconstructions take a few milliseconds to perform on a desktop gaming PC.
8 Theoretical considerations

The joint task adapted reconstruction defined in (21) is given by combining two optimal decision rules into a single decision rule, one for reconstruction that acts on data and the other encoding a task that acts on model parameters. It is therefore natural to investigate whether the theoretical machinery developed for Bayesian inversion can be used to analyze regularizing properties of this joint approach. An example would be to investigate the conditions under which the joint approach is a regularization in the formal sense, which means proving existence, stability, and posterior consistency that is preferably complemented by providing contraction rates, see [30, chapters 6-9] for the precise definitions.

Much of the theory on Bayesian inversion that deals with such matters is well understood for linear problems in the finite dimensional setting [48], but things quickly become complicated for infinite dimensional non-parametric problems. There has been nice progress recently on consistency, posterior contraction rates, and characterization of the microscopic fluctuations of the posterior that is relevant for Bayesian inversion, see [88, 18, 73] for nice surveys and [71] for a in-depth treatment of reconstruction relevant for tomographic imaging. On the other hand, the theory and associated results require too many restrictive assumptions that renders them inapplicable for analyzing the task adapted approach in (21). To conclude, theory of Bayesian inversion is in its current state not useful for characterizing conditions for when the joint task adapted reconstruction in (21) is a regularization.

Another line of investigation considers the potential advantage that comes with using a joint approach over a sequential one. Since the reconstruction and task operators are trained separately in a sequential approach, some information is inevitably lost when applying a regularized reconstruction operator. In contrast, both reconstruction and task operators are trained simultaneously in a joint approach so there is a better chance of preserving the information. Hence, we expect a joint approach to perform better, which is also supported by the observation in (22) and the empirical evidence in section 7.3.

Now, albeit convincing, the above heuristic argument is flawed! In fact, as stated by proposition 1, it is surprisingly difficult to theoretically prove that a joint approach outperforms a sequential one in a non-parametric setting where one has access to all of data. The reason is that many standard operators that map data space to model parameter space are formally information conserving in such a setting. The adjoint of the forward operator, its Moore-Penrose pseudo-inverse, and even some regularized reconstruction operators such as the usual Tikhonov regularization are information conserving under standard Gaussian noise. For 2D parallel beam tomography, yet another example is the filtered backprojection (FBP) reconstruction operator (with a filter that is strictly non-zero in frequency space).

**Proposition 1.** Let \( x \) be a \( X \)-valued random variable, and \( y \) be a \( Y \)-valued random variable, both defined on the same probability space. Let \( \Pi : Y \to Y \) be a measurable operator with closed range. Let \( B \) be an arbitrary measurable map defined on \( Y \) that
is injective when restricted to \( \text{ran}(\Pi) \). Then, the following holds:
\[
\mathbb{E}[f(x)|y] = \mathbb{E}[f(x) | \mathcal{B}(\Pi y)] \quad \text{for all } f
\quad \iff \quad x \perp \perp (y - \Pi y) | \Pi y
\]
where \( f \) spans all random variables over \( X \).

Before getting to the proof, let us comment on the implication of the statement above. The operator \( \Pi \) typically represents an orthogonal projection onto the closure of the range of \( A \). The result above states that the probability of \( x \) conditioned on \( y \) is the same as the one conditioned on \( \mathcal{B}(\Pi y) \) if and only if, given the knowledge of \( \Pi y \), the “noise” in the null space of \( \Pi \), namely \( \mathbb{E}[f(x)|y] - \mathbb{E}[f(x) | \mathcal{B}(\Pi y)] \), is independent of \( x \).

\textbf{Proof.} The proof is essentially a rewriting of the definitions. Introduce the notations \( y_1 := \Pi y \) and \( y_2 := y - \Pi y \), so that \( y = y_1 + y_2 \). Then \( \mathbb{E}[f(x)|y] = \mathbb{E}[f(x)|y_1,y_2] \), as \( y \) and \((y_1,y_2)\) generate the same \( \sigma \)-algebras. The injectivity of \( \mathcal{B} \) on \( \text{ran}(\Pi) \) implies that the \( \sigma \)-algebra generated by \( \mathcal{B} \circ \Pi \circ y \) and \( \Pi \circ y \) are the same, so \( \mathbb{E}[f(x)|y_1] = \mathbb{E}[f(x)|\mathcal{B}(y_1)] \). Now, requiring that \( \mathbb{E}[f(x)|y_1,y_2] = \mathbb{E}[f(x)|y_1] \) holds for all \( f \) is exactly the statement of conditional independence in the claim.

\textbf{Corollary 1.} Consider the setting in section 7.1 for task adapted reconstruction and assume in particular that \( y \) and \( z \) are conditionally independent given \( x \). Finally, let \( \mathcal{B} \) satisfy the assumptions in proposition 1; we also assume that the equality in proposition 1 holds, that is \( \pi(x | y) = \pi(x | \mathcal{B}(y)) \). Then,
\[
\pi(z | y) = \pi(z | \mathcal{B}(y)).
\]

\textbf{Proof.} The conditional independence assumption can be written as \( \pi(z | x, y) = \pi(z | x) \). Using this, we compute \( \pi(x,y,z) = \pi(z | x, y)\pi(x,y) = \pi(z | x)\pi(x,y) \), which yields
\[
\pi(x, z | y) = \pi(z | x)\pi(x | y).
\]

Notice now that \( \mathcal{B}(y) \) and \( z \) are also conditionally independent given \( x \), so we similarly obtain
\[
\pi(x, z | \mathcal{B}(y)) = \pi(z | x)\pi(x | \mathcal{B}(y)).
\]

Now, (32) and (33) imply in particular that
\[
\pi(z | y) = \int \pi(z, x | y) \, dx = \int \pi(z | x)\pi(x | y) \, dx
\]
\[
\pi(z | \mathcal{B}(y)) = \int \pi(z, x | \mathcal{B}(y)) \, dx = \int \pi(z | x)\pi(x | \mathcal{B}(y)) \, dx.
\]

Our assumption is that \( \pi(x | y) = \pi(x | \mathcal{B}(y)) \), which combined with (34) yields (31). This concludes the proof.
By corollary 1 we see directly that the conditional distribution of \( z \) given data \( y \in Y \) is, as \( \Delta \)-valued random variables, equal to the conditional distribution of \( z \) given an initial reconstruction \( B(y) \in X \). In particular, a task adapted reconstruction method (either sequential or joint) is equivalent to first performing reconstruction by applying the fixed operator \( B: Y \to X \), which is not trained, followed by \( C: X \to D \) that is given as

\[
C := T \circ A^\dagger \circ B^{-1}.
\]

Note here that \( C \), which is trained, is a measurable map defining a non-randomized decision rule that in principle serves as a “task” operator.

To summarize, we cannot resort to “information bottleneck” type of arguments as an explanation for why the joint approach should outperform only training a task operator in this general setting. On the other hand, the above argument hints that an explanation must involve either the choice of architecture or the training protocol. Both of these are examples of classical and widely studied problems in deep learning concerning why deep learning “works” and these remain largely unsolved. Another argument in favor of a joint approach is that it is highly non-trivial to select an appropriate architecture for parametrizing \( C \), whereas \( T \) and \( A^\dagger \) are easier to parametrize by means of neural networks. Another possible reason is that the operations, like evaluating \( B \) or its inverse \( B^{-1} \), may not be stable. Finally, as we have seen from the examples, using knowledge about the reconstruction may in fact act as a regularizer, either by improving the trainability or the generalization properties.

9 Discussion and outlook

A key aspect for the implementation of the joint task adapted reconstruction method in (21) is that both decision rules are given by trainable neural networks, which after joint training forms a single intertwined neural network. In such case, the problem reduces to solving (29).

The neural network for the reconstruction should here preferably incorporate knowledge about how data is generated. Learned iterative methods, like the Learned Primal-Dual method, are therefore well suited for this task since they are given by a (deep) neural network that embeds the forward operator and a statistical model for the noise in measured data [1, 2].

Next, as shown in sections 6.2 and 6.3, a wide range of tasks can be interpreted as applying an optimal decision rule on the model parameters. The abstract framework for task adapted reconstruction (section 7.1) works with any task that can be represented by a neural network as long as the parametrization and the loss functions are differentiable, like those listed in sections 6.2 and 6.3. Hence, our approach opens up for truly task adapted reconstruction that goes well beyond performing reconstruction jointly with simple feature extraction. In particular, more advanced tasks, such as image caption generation or image-processing steps in radiomics [31, 56], can be per-
formed jointly with reconstruction. This potential is also mentioned in the editorial for the special issue on machine learning for image reconstruction in IEEE Transactions on Medical Imaging [93] where the editors for the special issue introduce the term rawdiomics (on p. 1294) for task adapted reconstruction applied to radiomics.

An important advantage that comes with a joint approach is increased robustness. Advanced tasks, like radiomics, commonly rely on deep neural networks that are trained on images in a supervised setting. Images are however inferred in a pre-processing step from measured data, so contrast and texture may depend on the instrumentation used for acquiring the data and the reconstruction method used for computing the images. Hence, a neural network that has been trained against images acquired from a particular equipment, or obtained using a particular reconstruction method, may generalize poorly when either of these factors change. This is especially the case for tasks involving elements of visual classification, such as semantic segmentation, that can be sensitive to variations in texture and contrast. In contrast, task adapted reconstruction acts on measured data instead of images (model parameters). Using a reconstruction step that incorporates a physics guided model for how measured data is generated results in a joint approach that is much more robust against variations in how data is acquired and processed. As an example, jointly training a learned iterative method with neural network(s) involved in radiomics will result in a joint scheme that is expected to be much more robust against variations in scanner and acquisition protocol. This is essential if radiomics is to be part of clinical-decision support systems for improving diagnostic, prognostic, and predictive accuracy.

Another important advantage with the proposed task adapted reconstruction method relates to computationally feasibility. The trained neural network for task adapted reconstruction scales to large scale problems. Such scalability remains a serious issue with the variational approaches mentioned in section 4. As an example, state-of-the-art methods for joint reconstruction and segmentation are based on a variational approach using a the Mumford-Shah functional, which quickly become computationally unfeasible. In contrast, the 2D examples in fig. 4 for joint reconstruction and segmentation are obtained using the approach in section 7.3.2 and these take a few milliseconds on a desktop gaming PC.

Finally, examples involving tomographic image reconstruction (section 7.3) support the claim that a joint approach outperforms a sequential one. Understand this theoretically (section 8) is however an open problem. In particular, there is currently no theory motivating using a joint loss of the type in (20), even though empirical evidence suggests such a choice outperforms the en-to-end and sequential approaches.

Acknowledgments

The work by Jonas Adler, Olivier Verdier, and Ozan Öktem has been supported by the Swedish Foundation for Strategic Research grant AM13-0049, Industrial PhD
grant ID14-0055, and Elekta AB. Carola-Bibiane Schönlieb and Sebastian Lunz acknowledge support from the Engineering and Physical Sciences Research Council (EPSRC) “EP/K009745/1”, the EPSRC grant “EP/M00483X/1”, the EPSRC centre “EP/N014588/1”, the Leverhulme Trust project “Breaking the non-convexity barrier”, the CHiPS (Horizon 2020 RISE project grant), the Cantab Capital Institute for the Mathematics of Information, and the Alan Turing Institute “TU/B/000071”.

References


9 Discussion and outlook


9 Discussion and outlook


9 Discussion and outlook


9 Discussion and outlook


Learning to solve inverse problems using Wasserstein loss

Jonas Adler  Axel Ringh  Ozan Öktem  Johan Karlsson
KTH  KTH  KTH  KTH
Elekta

Published on NIPS Optimal Transport workshop 2017

Abstract
We propose using the Wasserstein loss for training in inverse problems. In particular, we consider a learned primal-dual reconstruction scheme for ill-posed inverse problems using the Wasserstein distance as loss function in the learning. This is motivated by miss-alignments in training data, which when using standard mean squared error loss could severely degrade reconstruction quality. We prove that training with the Wasserstein loss gives a reconstruction operator that correctly compensates for miss-alignments in certain cases, whereas training with the mean squared error gives a smeared reconstruction. Moreover, we demonstrate these effects by training a reconstruction algorithm using both mean squared error and optimal transport loss for a problem in computerized tomography.

1 Introduction
In inverse problems the goal is to determine model parameters from indirect noisy observations. Example of such problems arise in many different fields in science and engineering, e.g., in X-ray computed tomography (CT) [27], electron tomography [28], and magnetic resonance imaging [8]. Machine learning has recently also been applied in this area, especially in imaging applications. Using supervised machine-learning to solve inverse problems in imaging requires training data where ground truth images are paired with corresponding noisy indirect observations. The learning provides a mapping that associates observations to corresponding images. However, in several applications there are difficulties in obtaining the ground truth, e.g., in many cases it may have undergone a distortion. For example, a recent study showed that MRI images may be distorted by up to 4 mm due to, e.g., inhomogeneities in the main
magnetic field \[36\]. If these images are used for training, the learned MRI reconstruction will suffer in quality. Similar geometric inaccuracies arise in several other imaging modalities, such as Cone Beam CT and full waveform inversion in seismic imaging.

This work seeks to provide a scheme for learning a reconstruction scheme for an ill-posed inverse problem with a Wasserstein loss by leveraging upon recent advances in efficient solutions of optimal transport \([10, 22]\) and learned iterative schemes for inverse problems \([3]\). The proposed method is demonstrated on a computed tomography example, where we show a significant improvement compared to training the same network using mean squared error loss. In particular, using the Wasserstein loss instead of standard mean squared error gives a result that is more robust against potential miss-alignment in training data.

2 Background

2.1 Inverse problems

In inverse problems the goal is to reconstruct an estimate of the signal \(f_{\text{true}} \in X\) from noisy indirect measurements (data) \(g \in Y\) assuming

\[
g = T(f_{\text{true}}) + \delta g. \tag{1}
\]

In the above \(X\) and \(Y\) are referred to as the reconstruction and data space, respectively. Both are typically Hilbert or Banach spaces. Moreover \(T : X \rightarrow Y\) denotes the forward operator, which models how a given signal gives rise to data in absence of noise. Finally, \(\delta g \in Y\) is the noise component of data. Many inverse problems of interest are ill-posed, meaning that there is no unique solution to (1) and hence there is no inverse to \(T\). Typically reconstructions of \(f_{\text{true}}\) are sensitive to the data and small errors gets amplified. One way to mitigate these effects is to use regularization \([12]\).

Variational regularization In variational regularization one formulates the reconstruction problem as an optimization problem. To this end, one introduces a data discrepancy functional \(f \mapsto \mathcal{L}(T(f), g)\), where \(\mathcal{L} : Y \times Y \rightarrow \mathbb{R}\), that quantifies the miss-fit in data space, and a regularization functional \(S : X \rightarrow \mathbb{R}\) that encodes a priori information about \(f_{\text{true}}\) by penalizing undesirable solutions. For a given \(g \in Y\), this gives an optimization problem of the form

\[
\min_{f \in X} \mathcal{L}(T(f), g) + \lambda S(f). \tag{2}
\]
Here, $\lambda$ acts as a trade-off parameter between the data discrepancy and regularization functional. In many cases $\mathcal{L}$ is taken to be the negative data log-likelihood, e.g., $\mathcal{L}(T(f), g) = \|T(f) - g\|_2^2$ in the case of additive white Gaussian noise. Moreover, a typical choice for regularization functional is total variation (TV) regularization, $S(f) = \|\nabla f\|_1$ [33]. These regularizers typically give rise to large scale and non-differentiable optimization problems, which requires advanced optimization algorithms.

**Learning for inverse problems** In many applications, and so also in inverse problems, data driven approaches have shown dramatic improvements over the state-of-the-art [24]. Using supervised learning to solve an inverse problem amounts to finding a parametrized operator $T_{\Theta}^\dagger: Y \rightarrow X$ where the parameters $\Theta$ are selected so that

$$g = T(f_{\text{true}}) + \delta g \implies T_{\Theta}^\dagger(g) \approx f_{\text{true}}.$$ 

For inverse problems in image processing, such as denoising and deblurring, we have $Y = X$ and it is possible to apply a wide range of widely studied machine learning techniques, such as fully convolutional deep neural networks with various architectures, including fully convolutional networks [18] and denoising auto-encoders [38].

However, in more complicated inverse problems as in tomography, the data and reconstruction spaces are very different, e.g., their dimension after discretization may differ. For this reason, learning a mapping from $Y$ to $X$ becomes nontrivial, and classical architectures that map, e.g., images to images using convolutional networks cannot be applied as-is. One solution is to use fully-connected layers as in [30] for very small scale tomographic reconstruction problems. A major disadvantage with such a fully learned approach is that the parameters space has to be very high dimensional in order to be able to learn both the prior and the data model, which often renders it infeasible due to training time and lack of training data.

A more successful approach is to first apply some crude reconstruction operator $T^\dagger: Y \rightarrow X$ and then use machine learning to post process the result. This separates the learning from the complications of mapping between spaces since the operator $T^\dagger$ can be applied off-line, prior to training. Such an approach has been demonstrated for tomographic reconstruction in [31, 37]. Its drawback for ill posed inverse problems is that information is typically lost by using $T^\dagger$, and this information cannot be recovered by post processing.

Finally, another approach is to incorporate the forward operator $T$ and its adjoint $T^*$ into the neural network. In these learned iterative schemes,
classical neural networks are interlaced with applications of the forward and backward operator, thus allowing for the learned reconstruction operator to work directly from data without having to learn the data model. For example, in [39] an alternating direction method of multipliers (ADMM)-like scheme for Fourier inversion is learned and [32] consider solving inverse problems typically arising in image restoration by a learned gradient-descent scheme. In [4] this later approach is shown to be applicable to large scale tomographic inversion. Finally, in [3] they apply learning in both spaces $X$ and $Y$, yielding a Learned Primal-Dual scheme, and show that it outperforms learned post-processing for reconstruction of medical CT images.

**Loss functions for learning** Once the $\Theta$ parametrization of $T_\Theta^\dagger$ is set, the parameters are typically chosen by minimization of some loss functional $L$. Without doubt, the most common loss function is the mean squared error, also called $L_2$ loss, given by

$$L(\Theta) = \mathbb{E}_{f,g} \left[ \| T_\Theta^\dagger (g) - f \|_2^2 \right].$$

(3)

It has however been noted that it is sub-optimal for imaging, and a range of other loss functions have been investigated. These include the classical $\ell_p$ norms and the structural similarity index (SSIM) [40], as well as more complex losses such as perceptual losses [20] and adversarial networks [26].

Recently, optimal mass transport has also been considered as loss function for classification [14] and generative models [5]. In this work we consider using optimal transport for training a reconstruction scheme for ill-posed inverse problems.

### 2.2 Optimal mass transport and Sinkhorn iterations

In optimal mass transport the aim is to transform one distribution into another by moving the mass in a way that minimizes the cost of the movement. For an introduction and overview of the topic, see, e.g., the monograph [35]. Lately, the area has attracted a lot of research [10, 11, 9] with applications to, e.g., signal processing [17, 15, 19, 13] and inverse problems [7, 22].

The optimal mass transport problem can be formulated as follows: let $\Omega \subset \mathbb{R}^d$ be a compact set, and let $\mu_0$ and $\mu_1$ be two measures, defined on $\Omega$, with the same total mass. Given a cost $c : \Omega \times \Omega \rightarrow \mathbb{R}^+$ that describes the cost for transporting a unit mass from one point to another, find a (mass preserving) transference plan $M$ that is as cheap as possible. Here, the transference plan characterizes how to move the mass of $\mu_0$ in order to deform it into $\mu_1$. Letting
Background

The transference plan be a nonnegative measure $dM$ on the space $\Omega \times \Omega$ yields a linear programming problem in the space of measures:

$$T(\mu_0, \mu_1) = \min_{dM \geq 0} \int_{(x_0, x_1) \in \Omega \times \Omega} c(x_0, x_1) dM(x_0, x_1)$$

subject to $\mu_0(x_0) dx_0 = \int_{x_0 \in \Omega} dM(x_0, x_1)$, $\mu_1(x_1) dx_1 = \int_{x_0 \in \Omega} dM(x_0, x_1)$.

Although this formulation is only defined for measures $\mu_0$ and $\mu_1$ with the same total mass, it can also be extended to handle measures with unbalanced masses [15, 9]. Moreover, under suitable conditions one can define the Wasserstein metrics $W_p$ using $T$, by taking $c(x_0, x_1) = d(x_0, x_1)^p$ for $p \geq 1$ and where $d$ is a metric on $\Omega$, and $W_p(\mu_0, \mu_1) := T(\mu_0, \mu_1)^{1/p}$ [35, Definition 6.1]. As the name indicates, $W_p$ is a metric on the set of nonnegative measures on $\Omega$ with fixed mass [35, Theorem 6.9], and $T$ is weak* continuous on this set. One important property is that $T$ (and thus also $W_p$) does not only compare objects point by point, as standard $L^p$ metrics, but instead quantifies how the mass is moved. This makes optimal transport natural for quantifying uncertainty and modelling deformations [19, 21].

One way to solve the optimal transport problem in applications is to discretize $\Omega$ and solve the corresponding finite-dimensional linear programming problem. In this setting the two measures are represented by point masses on the discretization grid, i.e., by two vectors $\mu_0, \mu_1 \in \mathbb{R}_+^n$ where the element $[\mu_k]_i$ corresponds to the mass in the point $x(i) \in \Omega$ for $i = 1, \ldots, n$ and $k = 0, 1$. Moreover, a transference plan is represented by a matrix $M \in \mathbb{R}_+^{n \times n}$ where the value $m_{ij} := [M]_{ij}$ denotes the amount of mass transported from point $x(i)$ to $x(j)$. The associated cost of a transference plan is $\sum_{i,j=1}^n c_{ij} m_{ij} = \text{trace}(C^T M)$, where $[C]_{ij} = c_{ij} = c(x(i), x(j))$ is the transportation cost from $x(i)$ to $x(j)$, and by discretizing the constraints we get that $M$ is a feasible transference plan from $\mu_0$ to $\mu_1$ if the row sums of $M$ is $\mu_0$ and the column sums of $M$ is $\mu_1$. The discrete version of (4) thus takes the form

$$T(\mu_0, \mu_1) = \min_{M \geq 0} \text{trace}(C^T M)$$

subject to $\mu_0 = M 1_n$, $\mu_1 = M^T 1_n$.

where $M \geq 0$ denotes element-wise non-negativity of the matrix. However, even though (5) is a linear programming problem it is in many cases compu-
2 Background

tationally infeasible due to the vast number of variables. Since $M \in \mathbb{R}^{n \times n}_+$ the number of variables is $n^2$, and thus if one seek to solve the optimal transport problem between two $512 \times 512$ images this results in more than $6 \cdot 10^{10}$ variables.

One approach for addressing this problem was proposed by Cuturi [10] that introduces an entropic regularizing term $D(M) = \sum_{i,j=1}^{n}(m_{ij} \log(m_{ij}) - m_{ij} + 1)$ for approximating the transference plan, so the resulting perturbed optimal transport problem reads as

$$\min_{M \geq 0} \text{trace}(C^T M) + \varepsilon D(M)$$

subject to $\mu_0 = M 1_n$ $\mu_1 = M^T 1_n$.

One can show that an optimal solution to (6) is of the form

$$M = \text{diag}(u) K \text{diag}(v),$$

where $K = \exp(-C/\varepsilon)$ (point-wise exponential) is known, and $u, v \in \mathbb{R}_n^+$ are unknown. This shows that the solution is parameterized by only $2n$ variables. Moreover, the two vectors can be computed iteratively by so called Sinkhorn iterations, i.e., alternatingly compute $u$ and $v$ that matches $\mu_0$ and $\mu_1$ respectively. This is summarized in Algorithm 1 where $\odot$ denotes elementwise multiplication and $./$ elementwise division. The procedure has been shown to have a linear convergence rate, see [10] and references therein.

Moreover, when the underlying cost $c(x_0, x_1)$ is translation invariant the discretized cost matrix $C$, and thus also the transformation $K$, gets a Toeplitz-block-Toeplitz structure. This structure can be used in order to compute $Kv$ and $K^Tu$ efficiently using the fast Fourier transform in $O(n \log n)$, instead of naive matrix-vector multiplication in $O(n^2)$ [22]. This is crucial for applications in imaging since for images of size $512 \times 512$ pixels one would have to explicitly store and multiply with matrices of size $262144 \times 262144$.

Algorithm 1 Sinkhorn iterations for computing entropy-regularized optimal transport [10]

1: Input $C, \varepsilon, \mu_0, \mu_1$
2: initialize $v_0 > 0$ and $K = \exp(-C/\varepsilon)$
3: for $i = 1, \ldots, N$ do
4: $u_i \leftarrow \mu_0/(Kv_{i-1})$
5: $v_i \leftarrow \mu_1/(K^Tu_i)$
6: return $u_N^T (K \odot C)v_N$
3 Learning a reconstruction operator using Wasserstein loss

In this work we propose to use entropy regularized optimal transport (6) to train a reconstruction operator, i.e., to select the parameters as

$$\Theta^* \in \arg \min_{\Theta} \mathbb{E}_{f,g} \left[ T(T^\dagger_\Theta(g), f) \right].$$

(8)

This should give better results when data $g$ is not aligned with the ground truth $f$. To see this, consider the case when $f$ is a point mass. In that case training the network with the $L^2$ loss (3) will (in the ideal case) result in a perfect reconstruction composed with a convolution that “smears” the reconstruction over the area of possible mis-alignment. On the other hand since optimal mass transport does not only compare objects point-wise, the network will (in the ideal case) learn a perfect reconstruction combined with a movement of the object to the corresponding barycenter (centroid) of the miss-alignment. These statements are made more precise in the following propositions. Formal definitions and proofs are deferred to the appendix.

**Proposition 1.** Let $g \in L^2(\mathbb{R}^n)$, let $\tau$ be a $\mathbb{R}^n$-valued random variable with probability measure $dP(t)$, and let $g_\tau(x) := g(x - \tau)$. Then there exists a function $f \in L^2(\mathbb{R}^n)$ that minimizes $\mathbb{E}_\tau[\|f - g_\tau\|^2]$, and this $f$ has the form

$$f(x) = (dP * g)(x) := \int_{\mathbb{R}^n} g(x - t) dP(t).$$

**Proposition 2.** Let $\delta(x)$ be the Dirac delta function on $\mathbb{R}^n$, let $\tau$ be a $\mathbb{R}^n$-valued random variable with probability measure $dP(t)$, and let $\delta_\tau(x) := \delta(x - \tau)$. Then

$$\mathbb{E}_\tau[T(\delta_\tau, \mu)] = \int_{\mathbb{R}^n} \left( \int_{\mathbb{R}^n} c(t, x) dP(t) \right) d\mu(x) \quad \text{for } \mu \geq 0 \text{ and } \int_{\mathbb{R}^n} d\mu(x) = 1$$

and $\mathbb{E}_\tau[T(\delta_\tau, \mu)] = \infty$ otherwise. Furthermore, finding a $\mu$ that minimizes $\mathbb{E}_\tau[T(\delta_\tau, \mu)]$ is equivalent to finding the global minimizers to $F(x)$. In particular, if (i) the probability measure $dP$ is symmetric around its mean, (ii) the underlying cost $c$ is of the form $c(t, x) = d(x - t)$, where $d$ is convex and symmetric, and (iii) $dP$ and $d$ are such that

$$\int_{\mathbb{R}^n} d(x - t) dP(t) \quad \text{and} \quad \int_{\mathbb{R}^n} \partial d(x - t) dP(t)$$
both exist and are finite for all $x \in \mathbb{R}^n$, then $\mu(x) = \delta(x - \mathbb{E}[\tau])$ is an optimal solution. Furthermore, if $d$ is also strictly convex, then this is the unique minimizer.

To illustrate Propositions 1 and 2 we consider the following example. Let $\tau$ be uniformly distributed on $[-1, 1]$, and let $c(x_0, x_1) = (x_0 - x_1)^2$. This gives

$$F(x) = \frac{1}{2} \int_{-1}^{1} (x - t)^2 dt = \frac{1}{3} + x^2,$$

which has minimum $x = 0$, and hence the (unique) minimizer to $\mathbb{E}_\tau[T(\delta, \mu)]$ is $\mu(x) = \delta(x)$. For the $L_2$ case with the uniform distribution, the minimizer of $\mathbb{E}_\tau[\|f - g\|^2_2]$ is the smoothed function $g = \frac{1}{2} \chi_{[-1, 1]}$.

The most common choice of distance $c$ is to use the squared norm $c(x_0, x_1) = \|x_0 - x_1\|^2$, as in the previous example. In this case the result of Proposition 2 can be strengthened, as shown in the following example.

Let $\tau$ be a $\mathbb{R}^n$-valued random variable with probability measure $dP(t)$ with finite first and second moments, and let $c(x_0, x_1) = \|x_0 - x_1\|^2$. This gives

$$F(x) = \int_{\mathbb{R}^n} (x - t)^2 dP(t) = x^2 - 2x \mathbb{E}[\tau] + \mathbb{E}[\tau^2],$$

which has a unique global minimum in $x = \mathbb{E}[\tau]$ and hence $\mu(x) = \delta(x - \mathbb{E}[\tau]).$

### 4 Implementation and evaluation

We use the recently proposed learned primal-dual structure in [3] for learning a reconstruction operator $T^\top_\Theta$ for solving the inverse problem in (1). In this algorithm, a sequence of small blocks work alternatingly in the data (dual) space $Y$ and the reconstruction (primal) space $X$ and are connected using the forward operator $T$ and its adjoint $T^\top$. The algorithm works with any differentiable operator $T$, but we state the version for linear operators for simplicity in algorithm 2.
Algorithm 2 Learned Primal-Dual reconstruction algorithm

1: Initialize $f_0 \in X^{N_{\text{primal}}}, h_0 \in U^{N_{\text{dual}}}$
2: for $i = 1, \ldots, I$ do
3:   $h_i \leftarrow \Gamma_{\Theta^d}(h_{i-1}, \mathcal{T}(f_{i-1}^{(2)}), g)$
4:   $f_i \leftarrow \Lambda_{\Theta^p}(f_{i-1}, \mathcal{T}^*(h_{i}^{(1)}))$
5:   $\mathcal{T}^\dagger_{\Theta}(g) := f_I^{(1)}$

The method was implemented using ODL [2], ASTRA [34], and TensorFlow [1]. We used the reference implementation\(^1\) with default parameters, i.e., the number of blocks in the primal and dual space was $I = 10$, and the number of primal and dual variables was set to $N_{\text{primal}} = N_{\text{dual}} = 5$. Moreover, the blocks used a residual structure and had three layers of $3 \times 3$ convolutions with 32 filters. PReLU nonlinearities were used. Thus, this corresponds to a residual CNN with convolutional depth of $10 \cdot 2 \cdot 3 = 60$, as shown in graphical format in fig. 1. We used zero initial values, $f_0 = h_0 = 0$.

We compare a learned reconstruction operator of this form when trained using $L_2$ loss (3) and using optimal transport loss (8). Moreover, the evaluation is done on a problem similar to the evaluation problem in [3, 4], i.e., on a problem in computed tomography. More specifically, training is done on data that consists of randomly generated circles on a domain of $512 \times 512$ pixel,

\(^1\) https://github.com/adler-j/learned_primal_dual
4 Implementation and evaluation

(a) Phantom  →  (b) Translated phantom  →  (c) Data

Fig. 2: Example of data generation process used for training and validation, where 2a shows an example phantom, 2b is the phantom with a random translation and 2c is the data (sinogram) corresponding to 2b with additive white noise on top. The pair \((g_i, f_i) = (2c, 2a)\) is what is used in the training.

and the forward operator \(T\) is the ray transform \([27]\). What makes this an ill-posed problem is that the data acquisition is done from only 30 views with 727 parallel lines. Moreover, the source of noise is two-fold in this set-up: (i) the pairs \((g_i, f_i)\) of data sets and phantoms are not aligned, meaning that the data is computed from a phantom with a random change in position. This random change is independent for the different circles, and for each circle it is a shift which is uniformly distributed over \([-40, 40]\) pixels, both in up-down and left-right direction. (ii) on the data computed from the shifted phantom, 5\% additive Gaussian noise was added. For an example, see fig. 2.

The optimal mass transport distance computed with Sinkhorn iterations was used as loss function, where we used the transport cost

\[
c(x_1, x_2) = \left(1 - e^{-\|x_1 - x_2\|^4/80^4}\right).
\]

This was chosen since it heavily penalizes large movements, while not diverging to infinity which causes numerical instabilities. Moreover, \(c(x_1, x_2)^{1/4}\) is in fact a metric on \(\mathbb{R}^2\) (see lemma 4 in the appendix) and thus \(W_4(\mu_0, \mu_1) := T(\mu_0, \mu_1)^{1/4}\) gives rise to a Wasserstein metric on the space of images, where \(T(\mu_0, \mu_1)\) is the optimal mass transport distance with the transport cost \(c(x_1, x_2)\). Since this cost is translation invariant, the matrix-vector multiplications \(Ku\) and \(K^Tv\) can be done with fast Fourier transform, as mentioned in section 2.2, and this was implemented in Tensorflow. We used 10 Sinkhorn iterations with entropy regularization \(\varepsilon = 10^{-3}\), to approximate the optimal
Since the optimal mass transport function (6) is only finite for marginals $\mu_0$ and $\mu_1$ with the same total mass, in the training we normalize the output of the reconstruction $T^\dagger_{\Theta}(g)$ with mass($f$)/mass($T^\dagger_{\Theta}(g)$). This makes $T^\dagger_{\Theta}(g)$ invariant with respect to the total mass, which is undesirable. To compensate for this, a small penalization on the error in total mass was added to the loss function.

The training also followed [3] closely. In particular, we used $2 \cdot 10^4$ batches of size 1, using the ADAM optimizer [23] with default values except for $\beta_2 = 0.99$. The learning rate (step length) used was cosine annealing [25] with initial step length $10^{-3}$. Moreover, in order to improve training stability we performed gradient norm clipping [29] with norms limited to 1. The convolution parameters were initialized using Xavier initialization [16], and all biases were initialized to zero. The training took approximately 3 hours using a single Titan X GPU. The source code used to replicate these experiments are available online.

Results are presented in fig. 3. As can be seen, the reconstruction using $L^2$ loss “smears” the reconstruction to an extent where the shape is impossible to recover. On the other hand, the reconstruction using the Wasserstein loss retains the overall global shape of the object, although relative and exact positions of the circles are not recovered.

5 Conclusions and future work

In this work we have considered using Wasserstein loss to train a neural network for solving ill-posed inverse problems in imaging where data is not aligned with the ground truth. We give a theoretical motivation for why this should give better results compared to standard mean squared error loss, and demonstrate it on a problem in computed tomography. In the future, we hope that this method can be applied to other inverse problems and to other problems in imaging such as segmentation.

\[2 \text{https://github.com/adler-j/wasserstein_inverse_problems}\]
5 Conclusions and future work

Fig. 3: In 3a we show the validation phantom, which was generated from the same training set but not used in training, in 3b the translated phantom from which the validation data was computed, in 3c a reconstruction with neural network trained using mean squared error loss (3), and in 3d a reconstruction with neural network trained using optimal mass transport loss (8).

Appendix: Deferred definition and proofs

Proof of Proposition 1. To show that \( f(x) = (dP * g)(x) \in \mathcal{L}_2(\mathbb{R}^n) \) minimizes \( \mathbb{E}_\tau[\|f - g_r\|_2^2] \) we expand the expression and use Fubini’s theorem to get

\[
\mathbb{E}_\tau[\|f - g_r\|_2^2] = \int_{\mathbb{R}^n} \left( \int_{\mathbb{R}^n} |f(x) - g_x(x)|^2 \, dx \right) \, dP(t) = \\
\int_{\mathbb{R}^n} \left( \int_{\mathbb{R}^n} (f(x) - g_x(x))^2 \, dP(t) \right) \, dx.
\]

Rearranging terms and using that \( \int_{\mathbb{R}^n} dP(t) \, dt = 1 \), this can be written as

\[
\mathbb{E}_\tau[\|f - g_r\|_2^2] = \int_{\mathbb{R}^n} \left( f(x) - \int_{\mathbb{R}^n} g_x(x) \, dP(t) \right)^2 \, dx + c,
\]

where \( c \) is a constant. Using this it follows that the minimizing \( f \) is of the form

\[
f(x) = \int_{\mathbb{R}^n} g_t(x) \, dP(t).
\]
To see that $f \in L^2(\mathbb{R}^n)$ we note that, by using Fubini’s theorem, we have
\[
\|f\|^2_2 = \int_{\mathbb{R}^n} \left( \int_{\mathbb{R}^n} g_s(x)dP(t) \right)^2 dx = \int_{\mathbb{R}^n} \int_{\mathbb{R}^n} \left( \int_{\mathbb{R}^n} g_s(x)g_t(x)dx \right) dP(s)dP(t)
\leq \int_{\mathbb{R}^n} \int_{\mathbb{R}^n} \left( \frac{1}{2} \int_{\mathbb{R}^n} g_s(x)^2 + g_t(x)^2 dx \right) dP(s)dP(t) = \|g\|^2_2 < \infty
\]
where the first inequality is the arithmetic-geometric mean inequality. This completes the proof.

**Definition 3.** Let $h : \mathbb{R}^n \to \mathbb{R}$. A subgradient to $h$ in a point $y$ is a vector $v$ so that
\[
h(x) \geq h(y) + \langle v, x - y \rangle, \quad \forall x \in \mathbb{R}^n.
\]
The set of all subgradients in a point $y$ is called the subdifferential of $h$ at $y$, and is denoted by $\partial h(y)$. This is a set-valued operator, and for any measure $d\nu$ on $\mathbb{R}^n$ we define $(d\nu \ast \partial h)(y) := \int_{\mathbb{R}^n} \partial h(t)d\nu(y - t)$ to be the set-valued operator
\[
y \mapsto \left\{ \int_{\mathbb{R}^n} v(t)d\nu(y - t) \in \mathbb{R}^n \mid v(t) \in \partial h(t) \right\}.
\]

**Proof of Proposition 2.** We consider finding the marginal $\mu$ that minimize $\mathbb{E}_\tau[T(\delta_\tau, \mu)]$. Without loss of generality we assume that $\tau$ is zero-mean, since otherwise we simply consider $\tau - \mathbb{E}[\tau]$ which is a zero-mean random variable. First we note that $T(\delta_t, \mu)$ is only finite for nonnegative measures $\mu$ with total mass 1, and hence $\mathbb{E}_\tau[T(\delta_\tau, \mu)]$ is only finite for such measures. Second, for such a $\mu$ we have
\[
T(\delta_t, \mu) = \int_{\mathbb{R}^n} c(t, x)d\mu(x),
\]
since one needs to transport all mass in $\mu$ into the point $t$ where $\delta_t$ has its mass. Using this and expanding the expression for the expectation gives that
\[
\mathbb{E}_\tau[T(\delta_\tau, \mu)] = \int_{\mathbb{R}^n} T(\delta_t, \mu)dP(t) = \int_{\mathbb{R}^n} \left( \int_{\mathbb{R}^n} c(t, x)dP(t) \right) d\mu(x),
\]
where we have used Fubini’s theorem in the last step. This completes the first half of the statement.

To prove the second half of the statement, note that the optimal $\mu$ have support only in the global minimas of the function
\[
F(x) := \int_{\mathbb{R}^n} d(x - t)dP(t) = \int_{\mathbb{R}^n} d(t)dP(x - t),
\]
which by assumption exists and is finite. Now, since $d$ is convex we have that
\[
d(x) \geq d(y) + \langle \partial d(y), x - y \rangle, \quad \text{for all } x, y \in \mathbb{R}^n, \tag{9}
\]
and convolving this inequality with $dP$ gives the inequality
\[
F(x) \geq F(y) + \langle \int_{\mathbb{R}^n} \partial d(y-t)dP(t), x - y \rangle, \quad \text{for all } x, y \in \mathbb{R}^n, \tag{10}
\]
where all terms exist and are bounded by assumption. This shows that
\[
\int_{\mathbb{R}^n} \partial d(y-t)dP(t) \subset \partial F(y).
\]
Now, since $d$ is symmetric we have that $\partial d$ is anti-symmetric, i.e., that $\partial d(x) = -\partial d(-x)$, since
\[
d(x) = d(-x) \geq d(-y) + \langle \partial d(-y), -x + y \rangle = d(y) + \langle -\partial d(-y), x - y \rangle.
\]
Therefore
\[
\partial F(0) \supset \int_{\mathbb{R}^n} \partial d(-t)dP(t) \ni 0,
\]
where the last inclusion follows since $dP$ is symmetric and $\partial d$ is anti-symmetric. Now, since $0 \in \partial F(0)$ we have that $x = 0$ is a global minimizer to $F(x)$ [6, Theorem 16.2], and thus one optimal solution to the problem is $\mu(x) = \delta(x)$. Now, if $d$ is strictly convex, the inequality (9) is strict for $x \neq y$, and thus (10) is also strict, which shows that the optimal solution is unique. \qed

**Lemma 4.** Let $\| \cdot \|$ be a norm on $\mathbb{R}^m$. Then
\[
d(x_1, x_2) = (1 - e^{-\|x_1-x_2\|^n})^{\frac{1}{n}}.
\]
is a metric on $\mathbb{R}^m$ for $n \geq 1$.

**Proof.** It is easily seen that $d(x_1, x_2)$ is symmetric, nonnegative, and equal to zero if only if $x_1 = x_2$. Thus we only need to verify that the triangle inequality holds. To this end we note that if
\[
(1 - e^{-(a+b)n})^{\frac{1}{2n}} \leq (1 - e^{-a})^{\frac{1}{2}} + (1 - e^{-b})^{\frac{1}{2}}, \quad \text{for all } a, b \geq 0, \tag{11}
\]
for all $n \geq 1$, then by taking $a = \|x_1-x_2\|$, $b = \|x_2-x_3\|$, and using the triangle inequality for the norm $\| \cdot \|$ we have that
\[
d(x_1, x_3) = (1 - e^{-\|x_1-x_3\|^n})^{\frac{1}{2n}} \leq (1 - e^{-(\|x_1-x_2\|+\|x_2-x_3\|)n})^{\frac{1}{2n}}
\]
\[
\leq (1 - e^{-\|x_1-x_2\|^n})^{\frac{1}{2n}} + (1 - e^{-\|x_2-x_3\|^n})^{\frac{1}{2n}} = d(x_1, x_2) + d(x_2, x_3).
\]
Therefore we will show that (11) holds for all $n \geq 1$, and to do so we will
(i) show that if a function \( g : \mathbb{R}_+ \to \mathbb{R}_+ \) fulfills \( g(0) = 0, \ g(x)' \geq 0, \ g''(x) \leq 0 \) for all \( x \in \mathbb{R}_+ \), then \( g(x_1 + x_2) \leq g(x_1) + g(x_2) \).

(ii) show that for \( x \geq 0 \) the map \( x \mapsto (1 - e^{-x^n})^{\frac{1}{n}} \) fulfills the assumptions in (i) for any \( n \geq 1 \).

To show (i) we note that
\[
g(x_1 + x_2) = \int_0^{x_1 + x_2} g'(t)dt = \int_0^{x_1} g'(t)dt + \int_{x_1}^{x_1 + x_2} g'(t)dt \leq \int_0^{x_1} g'(t)dt + \int_0^{x_2} g'(t)dt = g(x_1) + g(x_2),
\]
where the inequality uses that \( g'(t) \geq g'(x + t) \) for any \( x, t \geq 0 \) since \( g''(x) \leq 0 \) for all \( x \geq 0 \).

To show (ii), let \( g(x) := (1 - e^{-x^n})^{\frac{1}{n}} \) and observe that \( g(0) = 0 \). Differentiating \( g \) twice gives
\[
g'(x) = \frac{e^{-x^n}(1 - e^{-x^n})^{\frac{1}{n}} x^{n-1}}{1 - e^{-x^n}} =: h_n(x^n),
\]
\[
g''(x) = -\left(1 - e^{-x^n}\right)^{\frac{1}{n}} x^{n-2} \frac{(ne^{x^n} x^n - x^n + e^{x^n} - ne^{x^n} + n - 1)}{(e^{x^n} - 1)^2}.
\]
For \( x \geq 0 \) we see that \( g'(x) \geq 0 \) for all \( n \geq 1 \). Moreover, for \( x \geq 0 \) we see that \( g''(x) \leq 0 \) for all \( x \geq 0 \) and for all \( n \geq 1 \) if and only if \( h_n(x^n) \geq 0 \). With the change of variable \( x^n = y \), we thus want to show that \( h_n(y) \geq 0 \) for all \( y \geq 0 \) and all \( n \geq 1 \). To see this we note that \( h_n(0) = 0 \) and that
\[
h'_n(y) = ne^y y + e^y - 1 \geq 0 \quad \text{for all } y \geq 0 \text{ and } n \geq 1.
\]
This shows (ii), and hence completes the proof.

References


5 Conclusions and future work


5 Conclusions and future work


5 Conclusions and future work


Banach Wasserstein GAN

Jonas Adler
Department of Mathematics
KTH - Royal institute of Technology
Research and Physics
Elekta
jonasad@kth.se

Sebastian Lunz
Department of Applied Mathematics
and Theoretical Physics
University of Cambridge
lunz@math.cam.ac.uk

Abstract

Wasserstein Generative Adversarial Networks (WGANs) can be used to generate realistic samples from complicated image distributions. The Wasserstein metric used in WGANs is based on a notion of distance between individual images, which induces a notion of distance between probability distributions of images. So far the community has considered $\ell^2$ as the underlying distance. We generalize the theory of WGAN with gradient penalty to Banach spaces, allowing practitioners to select the features to emphasize in the generator. We further discuss the effect of some particular choices of underlying norms, focusing on Sobolev norms. Finally, we demonstrate a boost in performance for an appropriate choice of norm on CIFAR-10 and CelebA.

1 Introduction

Generative Adversarial Networks (GANs) are one of the most popular generative models [6]. A neural network, the generator, learns a map that takes random input noise to samples from a given distribution. The training involves using a second neural network, the critic, to discriminate between real samples and the generator output.

In particular, [2, 7] introduces a critic built around the Wasserstein distance between the distribution of true images and generated images. The Wasserstein distance is inherently based on a notion of distance between images which in all implementations of Wasserstein GANs (WGAN) so far has been the $\ell^2$ distance. On the other hand, the imaging literature contains a wide range of metrics used to compare images [4] that each emphasize different features of interest, such as edges or to more accurately approximate human observer perception of the generated image.

There is hence an untapped potential in selecting a norm beyond simply the classical $\ell^2$ norm. We could for example select an appropriate Sobolev space to either emphasize edges, or large scale behavior. In this work we extend the classical WGAN theory to work on these and more general Banach spaces.

Our contributions are as follows:

- We introduce Banach Wasserstein GAN (BWGAN), extending WGAN implemented via a gradient penalty (GP) term to any separable complete normed space.
- We describe how BWGAN can be efficiently implemented. The only practical difference from classical WGAN with gradient penalty is that the $\ell^2$ norm is replaced with a dual norm. We also give theoretically grounded heuristics for the choice of regularization parameters.
- We compare BWGAN with different norms on the CIFAR-10 and CelebA datasets. Using the Space $L^{10}$, which puts strong emphasize on outliers, we achieve an unsupervised inception score of 8.31 on CIFAR-10, state of the art for non-progressive growing GANs.

2 Background

2.1 Generative adversarial networks

Generative Adversarial Networks (GANs) [6] perform generative modeling by learning a map \( G : Z \rightarrow B \) from a low-dimensional latent space \( Z \) to image space \( B \), mapping a fixed noise distribution \( P_r \) to a distribution of generated images \( P_G \).

In order to train the generative model \( G \), a second network \( D \) is used to discriminate between original images drawn from a distribution of real images \( P_r \) and images drawn from \( P_G \). The generator is trained to output images that are conceived to be realistic by the critic \( D \). The process is iterated, leading to the famous minimax game [6] between generator \( G \) and critic \( D \):

\[
\min_G \max_D \mathbb{E}_{X \sim P_r} [\log(D(X))] + \mathbb{E}_{Z \sim P_Z} [\log(1 - D(G(Z)))].
\]

Assuming the discriminator is perfectly trained, this gives rise to the Jensen–Shannon divergence (JSD) as distance measure between the distributions \( P_G \) and \( P_r \) [6, Theorem 1].

2.2 Wasserstein metrics

To overcome undesirable behavior of the JSD in the presence of singular measures [1], in [2] the Wasserstein metric is introduced to quantify the distance between the distributions \( P \) and \( Q \) on a Polish (e.g. separable completely metrizable) space \( X \). In its primal formulation, the Wasserstein-\( p \), \( p \geq 1 \), distance is defined as

\[
\text{Wass}_p(P,G) := \left( \inf_{\pi \in \Pi(P,G)} \mathbb{E}_{(X_1,X_2) \sim \pi} d_B(X_1,X_2)^p \right)^{1/p},
\]

where \( \Pi(P,G) \) denotes the set of distributions on \( B \times B \) with marginals \( P \) and \( G \). The Wasserstein distance is hence highly dependent on the choice of metric \( d_B \).

The Kantorovich-Rubinstein duality [19, 5.10] provides a way of more efficiently computing the Wasserstein-1 distance (which we will henceforth simply call the Wasserstein distance). \( \text{Wass} = \text{Wass}_1 \) between measures on high dimensional spaces. The duality holds in the general setting of Polish spaces and states that

\[
\text{Wass}(P,G) = \sup_{\|f\|_{Lip} \leq 1} \mathbb{E}_{X \sim P_r} f(X) - \mathbb{E}_{X \sim P_g} f(X).
\]

The supremum is taken over all Lipschitz continuous functions \( f : B \rightarrow \mathbb{R} \) with Lipschitz constant \( \leq 1 \). We note that in this dual formulation, the dependence of \( f \) on the choice of metric is encoded in the condition of \( f \) being 1-Lipschitz and recall that a function \( f : B \rightarrow \mathbb{R} \) is \( \gamma \)-Lipschitz if

\[
|f(x) - f(y)| \leq \gamma d_B(x,y).
\]

In an abstract sense, the Wasserstein metric could be used in GAN training by using a critic \( D \) to approximate the supremum in (3). The generator uses the loss \( \mathbb{E}_{Z \sim P_Z} D(G(Z)) \). In the case of a perfectly trained critic \( D \), this is equivalent to using the Wasserstein loss \( \text{Wass}(P_G, P_r) \) to train \( G \) [2, Theorem 3].

2.3 Wasserstein GAN

Implementing GANs with the Wasserstein metric requires to approximate the supremum in (3) with a neural network. In order to do so, the Lipschitz constraint has to be enforced on the network. In the paper Wasserstein GAN [2] this was achieved by restricting all network parameters to lie within a predefined interval. This technique typically guarantees that the network is \( \gamma \) Lipschitz for some \( \gamma \) for any metric space. However, it typically reduces the set of admissible functions to a proper subset...
of all $\gamma$ Lipschitz functions, hence introducing an uncontrollable additional constraint on the network. This can lead to training instabilities and artifacts in practice [7].

In [7] strong evidence was presented that the condition can better be enforced by working with another characterization of $1$–Lipschitz functions. In particular, they prove that if $B = \mathbb{R}^n$, $d(x, y)_B = \|x - y\|_2$ we have the gradient characterization

$$f \text{ is } 1 \text{–Lipschitz } \iff \|\nabla f(x)\|_2 \leq 1 \text{ for all } x \in \mathbb{R}^n.$$  

They softly enforce this condition by adding a penalty term to the loss function of $D$ that takes the form

$$\mathbb{E}_X \left( \|\nabla D(X)\|_2 - 1 \right)^2,$$  (4)

where the distribution of $X$ is taken to be the uniform distributions on lines connecting points drawn from $P_C$ and $P_r$.

However, penalizing the $\ell^2$ norm of the gradient corresponds specifically to choosing the $\ell^2$ norm as underlying distance measure on image space. Some research has been done on generalizing GAN theory to other spaces [18, 11], but in its current form WGAN with gradient penalty does not extend to arbitrary choices of underlying spaces $B$. We shall give a generalization to a large class of spaces, the (separable) Banach spaces, but first we must introduce some notation.

2.4 Banach spaces of images

A vector space is a collection of objects (vectors) that can be added together and scaled, and can be seen as a generalization of the Euclidean space $\mathbb{R}^n$. If a vector space $B$ is equipped with a notion of length, a norm $\| \cdot \|_B : B \to \mathbb{R}_+$, we call it a normed space. The most commonly used norm is the $\ell^2$ norm defined on $\mathbb{R}^n$, given by

$$\|x\|_2 = \left( \sum_{i=1}^n x_i^2 \right)^{1/2}.$$  

Such spaces can be used to model images in a very general fashion. In a pixelized model, the image space $B$ is given by the discrete pixel values, $B \sim \mathbb{R}^{n \times n}$. Continuous image models that do not rely on the concept of pixel discretization include the space of square integrable functions over the unit square. The norm $\| \cdot \|_B$ gives room for a choice on how distances between images are measured. The Euclidean distance is a common choice, but many other distance notions are possible that account for more specific image features, like the position of edges in Sobolev norms.

A normed space is called a Banach space if it is complete, that is, Cauchy sequences converge. Finally, a space is separable if there exists some countable dense subset. Completeness is required in order to ensure that the space is rich enough for us to define limits whereas separability is necessary for the usual notions of probability to hold. These technical requirements formally hold in finite dimensions but are needed in the infinite dimensional setting. We note that all separable Banach spaces are Polish spaces and we can hence define Wasserstein metrics on them using the induced metric $d_B(x, y) = \|x - y\|_B$.

For any Banach space $B$, we can consider the space of all bounded linear functionals $B^* \to \mathbb{R}$, which we will denote $B^*$ and call the (topological) dual of $B$. It can be shown [17] that this space is itself a Banach space with norm $\| \cdot \|_{B^*}$ given by

$$\|x^*\|_{B^*} = \sup_{x \in B} \frac{x^*(x)}{\|x\|_B},$$  (5)

In what follows, we will give some examples of Banach spaces along with explicit characterizations of their duals. We will give the characterizations in continuum, but they are also Banach spaces in their discretized (finite dimensional) forms.

$L^p$-spaces. Let $\Omega$ be some domain, for example $\Omega = [0, 1]^2$ to model square images. The set of functions $x: \Omega \to \mathbb{R}$ with norm

$$\|x\|_{L^p} = \left( \int_\Omega |x(t)|^p dt \right)^{1/p}$$  (6)

...
is a Banach space with dual $[L^p]^* = L^q$ where $1/p + 1/q = 1$. In particular, we note that $[L^2]^* = L^2$. The parameter $p$ controls the emphasis on outliers, with higher values corresponding to a stronger focus on outliers. In the extreme case $p = 1$, the norm is known to induce sparsity, ensuring that all but a small amount of pixels are set to the correct values.

**Sobolev spaces.** Let $\Omega$ be some domain, then the set of functions $x : \Omega \to \mathbb{R}$ with norm

$$
\| x \|_{W^{1,p}} = \left( \int_{\Omega} (|x(t)|^p + |
abla x(t)|^p dt) \right)^{1/p}
$$

where $\nabla x$ is the spatial gradient, is an example of a Sobolev space. In this space, more emphasis is put on the edges than in e.g. $L^p$ spaces, since if $\| x_1 - x_2 \|_{W^{1,p}}$ is small then not only are their absolute values close, but so are their edges.

Since taking the gradient is equivalent to multiplying with $\xi$ in the Fourier space, the concept of Sobolev spaces can be generalized to arbitrary (real) derivative orders $s$ if we use the norm

$$
\| x \|_{W^{s,p}} = \left( \int_{\Omega} \left( |\xi|^2 |F_x(t)|^p dt \right)^{1/p} \right)^{1/p},
$$

where $F$ is the Fourier transform. The tuning parameter $s$ allows to control which frequencies of an image are emphasized: A negative value of $s$ corresponds to amplifying low frequencies, hence prioritizing the global structure of the image. On the other hand, high values of $s$ amplify high frequencies, thus putting emphasis on sharp local structures, like the edges or ridges of an image.

The dual of the Sobolev space, $[W^{s,p}]^*$, is $W^{-s,q}$ where $q$ is as above [3]. Under weak assumptions on $\Omega$, all Sobolev spaces with $1 \leq p < \infty$ are separable. We note that $W^{0,1} = L^p$ and in particular we recover as an important special case $W^{0,2} = L^2$.

There is a wide range of other norms that can be defined for images, see appendix A and [5, 3] for a further overview of norms and their respective duals.

### 3 Banach Wasserstein GANs

In this section we generalize the loss (4) to separable Banach spaces, allowing us to effectively train a Wasserstein GAN using arbitrary norms.

We will show that the characterization of $\gamma$-Lipschitz functions via the norm of the differential can be extended from the $\ell_2$ setting in (4) to arbitrary Banach spaces by considering the gradient as an element in the dual of $B$. In particular, for any Banach space $B$ with norm $\| \cdot \|_B$, we will derive the loss function

$$
L = \frac{1}{\gamma} \left( \mathbb{E}_{X \sim p_X} D(X) - \mathbb{E}_{Y \sim p_Y} D(Y) \right) + \lambda \mathbb{E}_X \left( \frac{1}{\| \partial D(\hat{X}) \|_B} - 1 \right)^2,
$$

where $\lambda, \gamma \in \mathbb{R}$ are regularization parameters, and show that a minimizer of this is an approximation to the Wasserstein distance on $B$.

#### 3.1 Enforcing the Lipschitz constraint in Banach spaces

Throughout this chapter, let $B$ denote a Banach space with norm $\| \cdot \|_B$ and $f : B \to \mathbb{R}$ a continuous function. We require a more general notion of gradient: The function $f$ is called Fréchet differentiable at $x \in B$ if there is a bounded linear map $\partial f(x) : B \to \mathbb{R}$ such that

$$
\lim_{\| h \|_B \to 0} \frac{1}{\| h \|_B} | f(x + h) - f(x) - [\partial f(x)](h) | = 0.
$$

The differential $\partial f(x)$ is hence an element of the dual space $B^*$. We note that the usual notion of gradient $\nabla f(x)$ in $\mathbb{R}^n$ with the standard inner product is connected to the Fréchet derivative via $[\partial f(x)](h) = \nabla f(x) \cdot h$.

The following theorem allows us to characterize all Lipschitz continuous functions according to the dual norm of the Fréchet derivative.
Lemma 1. Assume $f : B \to \mathbb{R}$ is Fréchet differentiable. Then $f$ is $\gamma$-Lipschitz if and only if
\[
\|\partial f(x)\|_{B^*} \leq \gamma \quad \forall x \in B.
\] (10)

Proof. Assume $f$ is $\gamma$-Lipschitz. Then for all $x, h \in B$ and $\epsilon > 0$
\[
[\partial f(x)](h) = \lim_{t \to 0} \frac{1}{\epsilon} f(x + \epsilon h) - f(x) \leq \lim_{t \to 0} \frac{\gamma \|h\|_B}{\epsilon} = \gamma \|h\|_B,
\]
hence by the definition of the dual norm, eq. (5), we have
\[
\|\partial f(x)\|_{B^*} = \sup_{h \in B} \frac{\|\partial f(x)(h)\|}{\|h\|_B} \leq \sup_{h \in B} \frac{\gamma \|h\|_B}{\|h\|_B} \leq \gamma.
\]

Now let $f$ satisfy (10) and let $x, y \in B$. Define the function $g : \mathbb{R} \to \mathbb{R}$ by
\[
g(t) = f(x(t)), \quad \text{where} \quad x(t) = tx + (1 - t)y.
\]
As $x(t + \Delta t) - x(t) = \Delta f(x - y)$, we see that $g$ is everywhere differentiable and
\[
g'(t) = [\partial f(x(t))](x - y).
\]
Hence
\[
|g'(t)| = |[\partial f(x(t))](x - y)| \leq \|\partial f(x(t))\|_{B^*} \|x - y\|_B \leq \gamma \|x - y\|_B,
\]
which gives
\[
|f(x) - f(y)| = |g(1) - g(0)| \leq \int_0^1 |g'(t)| \, dt \leq \gamma \|x - y\|_B,
\]
thus finishing the proof. \qed

Using lemma 1 we see that a $\gamma$-Lipschitz requirement in Banach spaces is equivalent to the dual norm of the Fréchet derivative being less than $\gamma$ everywhere. In order to enforce this we need to compute $\|\partial f(x)\|_{B^*}$. As shown in section 2.4, the dual norm can be readily computed for a range of interesting Banach spaces, but we also need to compute $\partial f(x)$, preferably using readily available automatic differentiation software. However, such software can typically only compute derivatives in $\mathbb{R}^n$ with the standard norm.

Consider a finite dimensional Banach space $B$ equipped by any norm $\| \cdot \|_B$. By Lemma 1, gradient norm penalization requires characterizing (e.g. giving a basis for) the dual $B^*$ of $B$. This can be a difficult for infinite dimensional Banach spaces. In a finite dimensional however setting, there is an isomorphism implicitly relies on the fact that a basis of $B$ can be chosen and can be mapped to the corresponding dual basis. This does not generalize to the infinite dimensional setting, but we hope that this is not a very limiting assumption in practice.

We note that we can write $f = g \circ \iota$ where $g : \mathbb{R}^n \to \mathbb{R}$ and automatic differentiation can be used to compute the derivative $\partial g(x)$ efficiently. Further, note that the chain rule yields
\[
\partial f(x) = \iota^*(\partial g(\iota(x))),
\]
where $\iota^* : \mathbb{R}^n \to B^*$ is the adjoint of $\iota$ which is readily shown to be as simple as $\iota$. This shows that computing derivatives in finite dimensional Banach spaces can be done using standard automatic differentiation libraries with only some formal mathematical corrections. In an implementation, the operators $\iota, \iota^*$ would be implicit.

In terms of computational costs, the difference between general Banach Wasserstein GANS and the ones based on the $L^2$ metric lies in the computation of the dual norm. By the chain rule, any computational step outside the calculation of this gradient is the same for any choice of underlying notion of distance. This in particular includes any forward pass or backpropagation step through the layers of the network used as discriminator. If there is an efficient framework available to compute the gradient of the dual norm, as in the case of the Fourier transform used for Sobolev spaces, the computational expenses hence stay essentially the same independent of the choice of norm.
3.2 Regularization parameter choices

The network will be trained by adding the regularization term
\[ \lambda \mathbb{E}_x \left( \frac{\|D(\hat{x})\|_{B^*} - 1}{\gamma} \right)^2. \]

Here, \( \lambda \) is a regularization constant and \( \gamma \) is a scaling factor controlling which norm we compute. In particular, \( D \) will approximate \( \gamma \) times the Wasserstein distance. In the original WGAN-GP paper \cite{liu2017wasserstein} and most following work \( \lambda = 10 \) and \( \gamma = 1 \), while \( \gamma = 750 \) was used in Progressive GAN \cite{mirza2014conditional}. However, it is easy to see that these values are specific to the \( \ell_2 \) norm and that we would need to re-tune them if we change the norm. In order to avoid having to hand-tune these for every choice of norm, we will derive some heuristic parameter choice rules that work with any norm.

For our heuristic, we will start by assuming that the generator is the zero-generator, always returning zero. Assuming symmetry of the distribution of true images \( \mathbb{P}_r \), the discriminator will then essentially be decided by a single constant \( f(x) = c\|x\|_{B^*} \) where \( c \) solves the optimization problem

\[ \min_{c \in \mathbb{R}} \mathbb{E}_{X \sim \mathbb{P}_r} \left[ -\frac{\|X\|_{B^*}}{\gamma} + \frac{\lambda (c - \gamma)^2}{\gamma^2} \right]. \]

By solving this explicitly we find

\[ c = \gamma \left( 1 + \frac{\mathbb{E}_{X \sim \mathbb{P}_r} \|X\|_{B^*}}{2\lambda} \right)^{-1}. \]

Since we are trying to approximate \( \gamma \) times the Wasserstein distance, and since the norm has Lipschitz constant 1, we want \( c \approx \gamma \). Hence to get a small relative error we need \( \mathbb{E}_{X \sim \mathbb{P}_r} \|X\|_{B^*} \ll 2\lambda \). With this theory to guide us, we can make the heuristic rule

\[ \lambda \approx \mathbb{E}_{X \sim \mathbb{P}_r} \|X\|_{B^*}. \]

In the special case of CIFAR-10 with the \( \ell_2 \) norm this gives \( \lambda \approx 27 \), which agrees with earlier practice (\( \lambda = 10 \)) reasonably well.

Further, in order to keep the training stable we assume that the network should be approximately scale preserving. Since the operation \( x \to \partial D(x) \) is the deepest part of the network (twice the depth as the forward evaluation), we will enforce \( \|x\|_{B^*} \approx \|\partial D(x)\|_{B^*} \). Assuming \( \lambda \) was appropriately chosen, we find in general (by lemma 1) \( \|\partial D(x)\|_{B^*} \approx \gamma \). Hence we want \( \gamma \approx \|x\|_{B^*} \). We pick the expected value as a representative and hence we obtain the heuristic

\[ \gamma \approx \mathbb{E}_{X \sim \mathbb{P}_r} \|X\|_{B^*}. \]

For CIFAR-10 with the \( \ell_2 \) norm this gives \( \gamma = \lambda \approx 27 \) and may explain the improved performance obtained in \cite{mirza2014conditional}.

A nice property of the above parameter choice rules is that they can be used with any underlying norm. By using these parameter choice rules we avoid the issue of hand-tuning further parameters when training using different norms.

4 Computational results

To demonstrate computational feasibility and to show how the choice of norm can impact the trained generator, we implemented Banach Wasserstein GAN with various Sobolev and \( L^p \) norms, applied to CIFAR-10 and CelebA \((64 \times 64\) pixels). The implementation was done in TensorFlow and the architecture used was a faithful re-implementation of the residual architecture used in \cite{radford2015unsupervised}, see appendix B. For the loss function, we used the loss eq. \( (8) \) with parameters according to section 3.2 and the norm chosen according to either the Sobolev norm eq. \( (7) \) or the \( L^p \) norm eq. \( (6) \). In the case of the Sobolev norm, we selected units such that \( |\xi| \leq 5 \). Following \cite{mirza2014conditional}, we add a small \( 10^{-5} \mathbb{E}_{X \sim \mathbb{P}_r} \|D(X)\|^2 \) term to the discriminator loss to stop it from drifting during the training.

For training we used the Adam optimizer \cite{kingma2014adam} with learning rate decaying linearly from \( 2 \cdot 10^{-4} \) to 0 over 100,000 iterations with \( \beta_1 = 0, \beta_2 = 0.9 \). We used 5 discriminator updates per generator update. The batch size used was 64. In order to evaluate the reproducibility of the results on CIFAR-10, we
followed this up by training an ensemble of 5 generators using SGD with warm restarts following [12]. Each warm restart used 10,000 generator steps. Our implementation is available online\footnote{\url{https://github.com/adler-j/bwgan}}. Some representative samples from the generator on both datasets can be seen in figs. 1 and 5. See appendix C for samples from each of the $W^{s,2}$ and $L^p$ spaces investigated as well as samples from the corresponding Fréchet derivatives.

For evaluation, we report Fréchet Inception Distance (FID)[8] and Inception scores, both computed from 50K samples. A high image quality corresponds to high Inception and low FID scores. On the CIFAR-10 dataset, both FID and inception scores indicate that negative $s$ and large values of $p$ lead to better image quality. On CelebA, the best FID scores are obtained for values of $s$ between $-1$ and 0 and around $p = 0$, whereas the training become unstable for $p = 10$. We further compare our CIFAR-10 results in terms of Inception scores to existing methods, see table 4. To the best of

\begin{table}[h]
\centering
\begin{tabular}{|l|c|}
\hline
Method & Inception Score \\
\hline
DCGAN [16] & 6.16 ± .07 \\
EBGAN [21] & 7.07 ± .10 \\
WGAN-GP [7] & 7.86 ± .07 \\
CT GAN [20] & 8.12 ± .12 \\
SNGAN [14] & 8.22 ± .05 \\
$W^{s,2}$-BWGAN & 8.26 ± .07 \\
$L^{10}$-BWGAN & 8.31 ± .07 \\
Progressive GAN [9] & 8.80 ± .05 \\
\hline
\end{tabular}
\end{table}
our knowledge, the inception score of $8.31 \pm 0.07$, achieved using the $L^{10}$ space, is state of the art for non-progressive growing methods. Our FID scores are also highly competitive, for CIFAR-10 we achieve $16.43$ using $L^{1}$. We also note that our result for $W^{0,2} = \ell^2$ is slightly better than the reference implementation, despite using the same network. We suspect that this is due to our improved parameter choices.

5 How about metric spaces?

Gradient norm penalization according to lemma 1 is only valid in Banach spaces, but a natural alternative to penalizing gradient norms is to enforce the Lipschitz condition directly (see [15]). This would potentially allow training Wasserstein GAN on general metric spaces by adding a penalty term of the form

$$E_{X,Y} \left[ \frac{(f(X) - f(Y))}{d_{B}(X,Y)} - 1 \right]^2.$$  \hspace{1cm} (12)

While theoretically equivalent to gradient norm penalization when the distributions of $X$ and $Y$ are chosen appropriately, this term is very likely to have considerably higher variance in practice.

For example, if we assume that $d$ is not bounded from below and consider two points $x, y \in M$ that are sufficiently close then a penalty term of the Lipschitz quotient as in (12) imposes a condition on the differential around $x$ and $y$ in the direction $(x - y)$ only, i.e. only $|\partial f(\tilde{x})(x - y)| \leq 1$ is ensured. In the case of two distributions that are already close, we will with high probability sample the difference quotient in a spatial direction that is parallel to the data, hence not exhausting the Lipschitz constraint, i.e. $|\partial f(\tilde{x})(x - y)| \ll 1$. Difference quotient penalization (12) then does not effectively enforce the Lipschitz condition. Gradient norm penalization on the other hand ensures this condition in all spatial directions simultaneously by considering the dual norm of the differential.
On the other hand, if $d$ is bounded from below the above argument fails. For example, Wasserstein GAN over a space equipped with the trivial metric

$$d_{\text{trivial}}(x, y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{else} \end{cases}$$

approximates the Total Variation distance [19]. Using the regularizer eq. (12) we get a slight variation of Least Squares GAN [13]. We do not further investigate this line of reasoning.

6 Conclusion

We analyzed the dependence of Wasserstein GANs (WGANs) on the notion of distance between images and showed how choosing distances other than the $\ell^2$ metric can be used to make WGANs focus on particular image features of interest. We introduced a generalization of WGANs with gradient norm penalization to Banach spaces, allowing to easily implement WGANs for a wide range of underlying norms on images. This opens up a new degree of freedom to design the algorithm to account for the image features relevant in a specific application.

On the CIFAR-10 and CelebA dataset, we demonstrated the impact a change in norm has on model performance. In particular, we computed FID scores for Banach Wasserstein GANs using different Sobolev spaces $W^{s,p}$ and found a correlation between the values of both $s$ and $p$ with model performance.

While this work was motivated by images, the theory is general and can be applied to data in any normed space. In the future, we hope that practitioners take a step back and ask themselves if the $\ell^2$ metric is really the best measure of fit, or if some other metric better emphasize what they want to achieve with their generative model.

Acknowledgments

The authors would like to acknowledge Peter Maass for bringing us together as well as important support from Ozan Öktem, Axel Ringh, Johan Karlsson, Jens Sjölund, Sam Power and Carola Schönlieb.

The work by J.A. was supported by the Swedish Foundation of Strategic Research grants AM13-0049, ID14-0055 and Elekta. The work by S.L. was supported by the EPSRC grant EP/L016516/1 for the University of Cambridge Centre for Doctoral Training, and the Cambridge Centre for Analysis. We also acknowledge the support of the Cantab Capital Institute for the Mathematics of Information.

References


A Some further Banach spaces

There is some algebra for how to form new Banach spaces from known spaces. Specifically we have the following constructions that the reader might find useful.

**Weighted spaces.** Let $B_1$ be some separable Banach space with norm $\| \cdot \|_{B_1}$, then we can construct another space $B_2$ with norm $\| f \|_{B_2} := \| Af \|_{B_1}$ where $A : B_2 \to B_1$ is a continuous linear bijection. It is straightforward to show that the dual space $B_2^*$ has norm $\| f^* \|_{B_2^*} = \| A^{-*} f^* \|_{B_1^*}$ where $A^{-*} : B_1^* \to B_2^*$ is the adjoint of the inverse of $A$. These weighted spaces could be used to focus on some feature of interest, e.g. focus especially on the red color channel or on some spatial region of the image, the center perhaps, that is more important.

**Product spaces.** Let $B_1, \ldots, B_n$ be Banach spaces and let $B = B_1 \times \cdots \times B_n$ be the product space with norm $\| (x_1, \ldots, x_n) \|_B = \left( \sum_{i=1}^{n} \| x_i \|^p_{B_i} \right)^{1/p}$, then the dual space has norm $\| (x_1^*, \ldots, x_n^*) \|_{B^*} = \left( \sum_{i=1}^{n} \| x_i^* \|^q_{B_i^*} \right)^{1/q}$ where $1/p + 1/q = 1$. These spaces could be used to explicitly model the color channels or even to model multi-modal data such as a generator outputting both an image and a caption.

B Network details

The implementation on CIFAR-10 faithfully follows the source code from [7]. It uses of residual blocks consisting of "nonlinearity + conv + nonlinearity + conv + residual connection" and mean-pooling/nearest neighbor interpolation as building blocks. The generator starts from a latent space of 128 normally distributed random numbers and applies a dense layer to 4x4 images and applies a residual block then an interpolation repeatedly until the resolution 32x32 is reached. Then, a nonlinearity followed by a 1x1 convolution with 3 output channels is applied in order to obtain the generated color images.

The discriminator goes the other way using pooling with a final spatial mean-pooling followed by a dense layer. We used ReLU nonlinearities, all convolutions uses 128 channels and we used batch normalization after the nonlinearities in the generator. Following, we used uniform He initialization for all convolutions except the residual connections which used uniform Xavier initialization.

The implementation for CelebA follows that for CIFAR-10, with an additional residual block for further up/downsampling added both in the generator and discriminator.

See [7] and/or our open source implementation for further details.

C Further samples

We give samples from each of the Sobolev spaces $W^{s,2}$ investigated in the paper. The qualitative results mirror those observed in section 4 with higher $s$ indicating higher gradients in the discriminator’s Fréchet derivative, thus indicating a focus on higher frequency content. We also show further examples along with the corresponding loss gradients for some $L^p$ spaces on both the CelebA and CIFAR-10 dataset.
Figure 7: Samples for all $W_{s,2}$-spaces investigated on CIFAR-10.

(a) $s = -2$
(b) $s = 0.0$
(c) $s = -\frac{3}{2}$
(d) $s = 0.5$
(e) $s = -1.0$
(f) $s = 1.0$
(g) $s = -0.5$
(h) $s = 2.0$
Figure 8: Fréchet derivatives for all \( W^{s,2} \)-spaces investigated on CIFAR-10.
Figure 9: Samples for all $W^{s,2}$-spaces investigated on CelebA.

(a) $s = -2$
(b) $s = 0.0$
(c) $s = -\frac{1}{2}$
(d) $s = 0.5$
(e) $s = -1.0$
(f) $s = 1.0$
(g) $s = -0.5$
(h) $s = 2.0$
Figure 10: Fréchet derivatives for all $W^{s,2}$-spaces investigated on CelebA.
Figure 11: Samples from all $L^p$-spaces investigated on CIFAR-10.
Figure 12: Fréchet derivatives for all $L^p$-spaces investigated on CIFAR-10.
Figure 13: Samples from all $L^p$-spaces investigated on CelebA.
Figure 14: Fréchet derivatives for all $L^p$-spaces investigated on CelebA.
Deep Bayesian Inversion

Jonas Adler  
KTH  
Elekta

Ozan ¨Oktem  
KTH

Preprint published 2018

Abstract

Characterizing statistical properties of solutions of inverse problems is essential for decision making. Bayesian inversion offers a tractable framework for this purpose, but current approaches are computationally unfeasible for most realistic imaging applications in the clinic. We introduce two novel deep learning based methods for solving large-scale inverse problems using Bayesian inversion: a sampling based method using a Wasserstein GAN with a novel mini-discriminator and a direct approach that trains a neural network using a novel loss function. The performance of both methods is demonstrated on image reconstruction in ultra low dose 3D helical CT. We compute the posterior mean and standard deviation of the 3D images followed by a hypothesis test to assess whether a “dark spot” in the liver of a cancer stricken patient is present. Both methods are computationally efficient and our evaluation shows very promising performance that clearly supports the claim that Bayesian inversion is usable for 3D imaging in time critical applications.

1 Introduction

In several areas of science and industry there is a need to reliably recover a hidden multidimensional model parameter from noisy indirect observations. A typical example is when imaging/sensing technologies are used in medicine, engineering, astronomy, and geophysics.

These inverse problems are often ill-posed, meaning that small errors in data may lead to large errors in the model parameter and there are several possible model parameter values that are consistent with observations. Addressing ill-posedness is critical in applications where decision making is based on the recovered model parameter, like in image guided medical diagnostics. Furthermore, many highly relevant inverse problems are large-scale; they involve large amounts of data and high-dimensional model parameter spaces.
**Bayesian inversion** Bayesian inversion is a framework for assigning probabilities to a model parameter given data (posterior) by combining a *data model* with a *prior model* (section 2). The former describes how measured data is generated from a model parameter whereas the latter accounts for information about the unknown model parameter that is known beforehand. Exploring the posterior not only allows for recovering the model parameter in a reliable manner by computing suitable estimators, it also opens up for a complete statistical analysis including quantification of the uncertainty.

A key part of Bayesian inversion is to express the posterior using Bayes’ theorem, which in turn requires access to the data likelihood, a prior, and a probability measure for data. The data likelihood is often given from insight into the physics of how data is generated (simulator). The choice of prior (section 2.1) is less obvious but important since it accounts for a priori information about the true model parameter. It is also very difficult to specify a probability distribution for data, which is required by many estimators. Finally, the computational burden associated with exploring the posterior (section 2.2) prevents usage of Bayesian inversion in most imaging applications.

To exemplify the above, consider clinical 3D computed tomography (CT) imaging where the model parameter represents the interior anatomy and data is x-ray radiographs taken from various directions. A natural prior in this context is that the object (model parameter) being imaged is a human being, but explicitly handcrafting such a prior is yet to be done. Instead, current priors prescribe roughness or sparsity, which suppresses unwanted oscillatory behavior at the expense of finer details. Next, the model parameter is typically $512^3$-dimensional and data is of at least same order of magnitude. Hence, exploring the posterior in a timely manner is challenging, e.g., uncertainty quantification in Bayesian inversion remains intractable for such large-scale inverse problems.

## 2 Statistical Approach to Inverse Problems

Uncertainty refers in general to the accuracy by which one can determine a model parameter. In an inverse problems, this rests upon the ability to explore the statistical distribution of model parameters given measured data. More precisely, the posterior probability of the model parameter conditioned on observed data describes all possible solutions to the inverse problem along with their probabilities [21, 19] and it is essential for uncertainty quantification.

Bayesian inversion uses Bayes’ theorem [19, Theorem 14] to characterize
the posterior:
\[ p(x \mid y) = \frac{p(x)p(y \mid x)}{p(y)}. \]
Here, \( p(y \mid x) \) is given by the data model that is usually derived from knowledge about how data is generated and \( p(x) \) is given by the prior model that represents information known beforehand about the true (unknown) model parameter.

A tractable property of Bayesian inversion is that small changes in data lead to small changes in the posterior even when the inverse problem is ill-posed in the classical sense [19, Theorem 16], so Bayesian inversion is stable. Different reconstructions can be obtained by computing different estimators from the posterior and there is also a natural framework for uncertainty quantification, e.g., by computing Bayesian credible sets.

The posterior is however quite complicated with no closed form expression, so much of the contemporary research focuses on realizing the aforementioned advantages with Bayesian inversion without requiring access to the full posterior, see [19] for a nice survey. Some related key challenges were mentioned earlier in the introduction; choosing a “good” prior, specifying the probability distribution of data, and to explore the posterior in a computationally feasible manner.

### 2.1 Choosing a prior model

The difficulty in selecting a prior model lies in capturing the relevant a priori information. Bayesian non-parametric theory [25] provides a large class of handcrafted priors, but these only capture a fraction of the a priori information that is available. Figure 1 illustrates this by showing random samples generated from priors commonly used by state-of-the-art approaches in image recovery [37, 15] as well as samples from typical clinical CT images. The handcrafted priors primarily encode regularity properties, like roughness or sparsity, and it would clearly be stretching our imagination to claim that corresponding samples represent natural images.

### 2.2 Computational feasibility

Exploring the posterior for inverse problems in imaging often leads to large scale numerics since this mounts to sampling from a high dimensional probability distribution. Most approaches, see section 6, are either not fast enough or rely on simplifying assumptions that does not hold in many applications.
For the above reasons, in large scale inverse problems one tends to reconstruct a single point estimate of the posterior distribution, the most common being the maximum a posteriori (MAP) estimator that corresponds to the most likely reconstruction given the data. A drawback that comes with working with single estimators is that these cannot include all the information present in the posterior distribution. It is clear that knowledge about the full posterior would have dramatic impact upon how solutions to inverse problems are intertwined into decision making. As an example, in medical imaging, practitioners would be able to compute the probability of a tumor being an image artifact, which in turn is necessary for image guided hypothesis testing.

3 Contribution

Our overall contribution is to suggest two generic, yet adaptable, frameworks for uncertainty quantification in inverse problems that are computationally feasible and where both the prior and probability distribution of data are given implicitly through supervised examples instead of being handcrafted. The approach is based on recent advances in generative adversarial networks (GANs) from deep learning and we demonstrate its performance on ultra low
Our main contribution is Deep Posterior Sampling (section 4.1) where generative models from machine learning are used to sample from a high-dimensional unknown posterior distribution in the context of Bayesian inversion. This is made possible by a novel conditional Wasserstein GAN (WGAN) discriminator (appendix C.2). The approach is generic and applies in principle to any inverse problem assuming there is relevant training data. It can be used for performing statistical analysis of the posterior on $X$, e.g., by computing various estimators.

Independently, we also introduce Deep Direct Estimation (section 4.2) where one directly computes an estimator using a deep neural network trained on a cleverly chosen loss (appendix C.3). Deep direct estimation is faster than posterior sampling, but it mainly applies to statistical analysis that is based on evaluating a pre-determined estimator. Both approaches should give similar quantitative results when used for evaluating the same estimator.

We demonstrate the performance and computational feasibility for ultra low dose CT imaging in a clinical setting by computing some estimators and performing a hypothesis test (section 5).

4 Deep Bayesian Inversion

As already stated, in Bayesian inversion both the model parameter $x$ and measured data $y$ are assumed to be generated by random variables $x$ and $y$, respectively. The ultimate goal is to recover the posterior $\pi(x | y)$, which describes all possible solutions $x = x$ along with their probabilities given data $y = y$.

We here outline two approaches that can be used to perform various statistical analysis on the posterior. Deep Posterior Sampling is a technique for learning how to sample from the posterior whereas Deep Direct Estimation learns various estimators directly.

4.1 Deep Posterior Sampling

The idea is to explore the posterior by sampling from a generator that is defined by a WGAN, which has been trained using a conditional WGAN discriminator.

To describe how a WGAN can be used for this purpose, let data $y \in Y$ be fixed and assume that $\pi(x \mid y)$, the posterior of $x$ at $y = y$, can be approximated by elements in a parametrized family $\{G_\theta(y)\}_{\theta \in \Theta}$ of probability measures on
X. The best such approximation is defined as \( G_{θ^*}(y) \) where \( θ^* ∈ Θ \) solves
\[
θ^* ∈ \arg \min_{θ ∈ Θ} \ell(G_θ(y), π(x \mid y)).
\] (1)

Here, \( \ell \) quantifies the “distance” between two probability measures on \( X \). We are however interested in the best approximation for “all data”, so we extend (1) by including an averaging over all possible data. The next step is to choose a distance notion \( \ell \) that desirable from both a theoretical and a computational point of view. As an example, the distance should be finite and computational feasibility requires using it to be differentiable almost everywhere, since this opens up for using stochastic gradient descent (SGD) type of schemes. The Wasserstein 1-distance \( \mathcal{W} \) (appendix A) has these properties [8] and sampling from the posterior \( π(x \mid y) \) can then be replaced by sampling from the probability distribution \( G_{θ^*}(y) \) where \( θ^* \) solves
\[
θ^* ∈ \arg \min_{θ ∈ Θ} E_{y \sim σ}[\mathcal{W}(G_θ(y), π(x \mid y))].
\] (2)

In the above, \( σ \) is the probability distribution for data and \( y \sim σ \) generates data.

Observe now that evaluating the objective in (2) requires access to the very posterior that we seek to approximate. Furthermore, the distribution \( σ \) of data is often unknown, so an approach based on (2) is essentially useless if the purpose is to sample from an unknown posterior. Finally, evaluating the Wasserstein 1-distance directly from its definition is not computationally feasible.

On the other hand, as shown in appendix C.1, all of these drawbacks can be circumvented by rewriting (2) as an expectation over the joint law \((x, y) \sim μ\). This makes use of specific properties of the Wasserstein 1-distance (Kantorovich-Rubenstein duality) and one obtains the following approximate version of (2):
\[
θ^* ∈ \arg \min_{θ ∈ Θ} \left\{ \sup_{φ ∈ Φ} E_{(x, y) \sim μ}[D_φ(x, y) - E_{z \sim η}[D_φ(G_θ(z), y)]] \right\}.
\] (3)

In the above, \( G_θ: Z \times Y → X \) (generator) is a deterministic mapping such that \( G_θ(z, y) \sim G_θ(y) \), where \( z \sim η \) is a ‘simple’ \( Z \)-valued random variable in the sense that it can be sampled in a computationally feasible manner. Next, the mapping \( D_φ: X \times Y → ℝ \) (discriminator) is a measurable mapping that is 1-Lipschitz in the \( X \)-variable.
On a first sight, it might be unclear why (3) is better than (2) if the aim is
to sample from the posterior, especially since the joint law $\mu$ in (3) is unknown.
The advantage becomes clear when one has access to supervised training data
for the inverse problem, i.e., i.i.d. samples $(x_1, y_1), \ldots, (x_m, y_m)$ generated by
the random variable $(x, y) \sim \mu$. The $\mu$-expectation in (3) can then be replaced
by an averaging over training data.

To summarize, solving (3) given supervised training data in $X \times Y$ amounts
to learning a generator $G_{\theta^*}(z, \cdot) : Y \rightarrow X$ such that $G_{\theta^*}(z, y)$ with $z \sim \eta$
is approximately distributed as the posterior $\pi(x \mid y)$. In particular, for given
$y \in Y$ we can sample from $\pi(x \mid y)$ by generating values of $z \mapsto G_{\theta^*}(z, y) \in X$
in which $z \in Z$ is generated by sampling from $z \sim \eta$.

An important part of the implementation is the concrete parameterizations
of the generator and discriminator:

$$G_{\theta} : Z \times Y \rightarrow X \quad \text{and} \quad D_{\phi} : X \times Y \rightarrow \mathbb{R}.$$  

We here use deep neural networks for this purpose and following [27], we
softly enforce the 1-Lipschitz condition on the discriminator by including a
gradient penalty term to the training objective in (3). Furthermore, if (3)
is implemented as is, then in practice $z$ is not used by the generator (so
called mode-collapse). To solve this problem, we introduce a novel conditional
mini-batch discriminator that can be used with conditional WGAN without
impairing upon its analytical properties (claim 1), see appendix C.2 for more
details.

### 4.2 Deep Direct Estimation

The idea here is to train a deep neural network to directly approximate an es-
timator of interest without resorting to generating samples from the posterior
as in posterior sampling (section 4.1).

Deep direct estimation relies on the well known result:

$$\mathbb{E}_{w}(w \mid y = \cdot) = \min_{h : Y \to W} \mathbb{E}_{(y, w)} \left[ \|h(y) - w\|_W^2 \right]. \quad (4)$$

In the above, $w$ is any random variable taking values in some measurable
Banach space $W$ and the minimization is over all $W$-valued measurable maps
on $Y$. See proposition 2 in appendix C.3 for a precise statement. This is useful
since many estimators relevant for uncertainty quantification are expressible
using terms of this form for appropriate choices of $w$.  

Specifically, appendix C.3 considers two (deep) neural networks \( T^\dagger_{\theta^*} : Y \rightarrow X \) and \( h_{\phi^*} : Y \rightarrow X \) with appropriate architectures that are trained according to

\[
\begin{align*}
\theta^* & \in \arg \min_{\theta} \left\{ \mathbb{E}_{(x,y) \sim \mu} \left[ \| x - T^\dagger_{\theta}(y) \|_X^2 \right] \right\} \\
\phi^* & \in \arg \min_{\phi} \left\{ \mathbb{E}_{(x,y) \sim \mu} \left[ \| h_{\phi}(y) - (x - T^\dagger_{\theta^*}(y)) \|^2_X \right] \right\}.
\end{align*}
\]

The resulting networks will then approximate the conditional mean and the conditional point-wise variance, respectively. Finally, if one has supervised training data \((x_i, y_i)\), then the joint law \(\mu\) above can be replaced by its empirical counterpart and the \(\mu\)-expectation is replaced by an averaging over training data.

As already indicated, by using (4) it is possible to re-write many estimators as minimizers of an expectation. Such estimators can then be approximated using the direct estimation approach outlined here. This should coincide with computing the same estimator by posterior sampling (section 4.1). Direct estimation is however significantly faster, but not as flexible as posterior sampling since each estimator requires a new neural network that specifically trained for that estimator. Section 5 compares outcome from both approaches.

5 Numerical Experiments

We evaluate the feasibility of posterior sampling (section 4.1) to sample from the posterior and Direct Estimation (section 4.2) to compute mean and point-wise variances for clinical 3D CT imaging.

5.1 Set-up

Our supervised data consists of pairs of 3D CT images \((x_i, y_i)\) generated by \((x, y)\) where \(x_i\) is a normal dose 3D image that serves as the ‘ground truth’ and \(y_i\) is the filtered back-projection (FBP) 3D reconstruction computed from ultra low dose CT data associated with \(x_i\).

One could here let \(y_i\) be the ultra low dose CT data itself, which results in more complex architectures of the neural networks. On the other hand, using FBP as a pre-processing step (i.e., \(y_i\) is FBP reconstruction from ultra low dose data) simplifies the choice of architectures and poses no limitation in the theoretical setting with infinite data (see [4, section 8]).
Training data  We used training data from the Mayo Clinic Low Dose CT challenge [47]. This data consists of ten CT scans, of which we use nine for training and one for evaluation. Each 3D image \( x_i \) has a corresponding ultra low dose data that is generated by using only \( \approx 10\% \) of the full data and adding additional Poisson noise so that the dose corresponds to 2\% of normal dose. Applying FBP on this data yields the ultra low dose CT images, see appendix D.1 for a detailed description.

An example of normal dose CT reconstruction, tomographic data, and the ultra low dose FBP reconstruction is shown in fig. 2.

Network architecture and training  The operators
\[
G_\theta: Z \times Y \rightarrow X \quad T_{\theta^*}: Y \rightarrow X \\
D_\phi: X \times Y \rightarrow \mathbb{R} \quad h_{\phi^*}: Y \rightarrow X
\]
are represented by multi-scale residual neural networks. For computational reasons, we applied the method slice-wise, see appendix D.2 for details regarding the exact choice of architecture and training procedure.

The parts related to the inverse problem (tomography) were implemented using the ODL framework [2] with ASTRA [62] as back-end for computing the ray-transform and its adjoint. The learning components were implemented in TensorFlow [1].
5.2 Results

Estimators A typical use-case of Bayesian inversion is to compute estimators from the posterior. In our case, we are interested in the conditional mean and point-wise standard deviation (square root of variance).

When using posterior sampling, we compute the conditional mean and point-wise standard deviations based on 1 000 images sampled from the posterior, see appendix B for some examples of such images. For direct estimation we simply evaluated the associated trained networks. Both approaches are computationally feasible, the time needed per slice to compute these estimators is 40 s using posterior sampling based on 1 000 samples and 80 ms for direct estimation.

The mean and standard-deviations that were computed using both methods are shown in fig. 3. We note that results from the methods agree very well with each other, indicating that the posterior samples follow the posterior quite well, or at least that the methods have similar bias. The posterior mean looks as one would expect, with highly smoothed features due to the high noise level. Likewise, the standard deviation is also as one would expect, with high uncertainties around the boundaries of the high contrast objects. We also note that the standard deviation at the white “blobs” that appear in some samples (see appendix B) is quite high, indicating that the model is uncertain about their presence. There is also a background uncertainty at ≈ 20 HU due to point-wise noise in the reference normal-dose scans that we take as ground truth.

Uncertainty quantification We here show how to use Bayesian credible sets for clinical image guided decision making. One computes a reconstruction from ultra low dose data (middle image in fig. 2), identifies one or more features, and then seeks to estimate the likelihood for the presence of these features.

Formalizing the above, let Δ denote the difference in mean intensity in the reconstructed image between a region encircling the feature and the surrounding organ, which in our example is the liver. The feature is said to “exist” whenever Δ is bigger than a certain threshold, say 10 HU.

To use posterior sampling, start by computing the conditional mean image (top left in fig. 3) by sampling from the posterior using the conditional WGAN approach in section 4.1. There is a dark “spot” in the liver (possible tumor) and a natural clinical question is to statistically test for the presence of this feature. To do this, compute Δ for a number of samples generated by posterior sampling, which here is the same 1 000 samples used for com-
5 Numerical Experiments

Fig. 3: Conditional mean and point-wise standard deviation (pStd) computed from test data (fig. 2) using posterior sampling (section 4.1) and direct estimation (section 4.2).

puting the conditional mean. We estimate $p := \text{Prob}(\Delta > 10 \text{ HU})$ from the resulting histogram in fig. 4 and clearly $p > 0.95$, indicating that the “dark spot” feature exists with at least 95% significance. This is confirmed by the ground truth image (left image in fig. 2). The conditional mean image also under-estimates $\Delta$, whose true value is the vertical line in fig. 4. This is to be expected since the prior introduces a bias towards homogeneous regions, a bias that decreases as noise level decreases.

To perform the above analysis using direct estimation, start with computing the conditional mean image from the same ultra-low dose data using direct estimation. As expected, the resulting image (top right in fig. 3) shows a dark “spot” in the liver. Now, designing and training a neural network that directly estimates the distribution of $\Delta$ is unfeasible in a general setting. However, as shown in section 4.2, this is possible if one assumes pixels are independent of each other. The estimated distribution of $\Delta$ is the curve in fig. 4 and we get $p > 0.95$, which is consistent with the result obtained using posterior sampling. The direct estimation approach is based on assuming independent pixels, so it will significantly underestimate the variance. In con-
6 Related Work

Deep learning based methods are increasingly used for medical image reconstruction, either by using deep learning for post-processing [38, 35] or by integrating deep learning into the image reconstruction [66, 5, 6, 29, 16, 28, 30]. These papers start by specifying the loss and then use a deep neural network to minimize the expected loss. This essentially amounts to directly computing a Bayes estimator with a risk is given by the loss. The loss is often the squared $L_2$-distance, which implicitly implies that one approximates the conditional mean. Hence, the above approaches could be seen as examples of deep direct estimation. There is however an important difference, in deep direct estimation one starts by explicitly specify the estimator, which then implies the appropriate loss function.

There has also been intense research in selecting a loss function different from the $L_2$-loss [36, 7] and specifically GAN-like methods have been applied to image post-processing in CT [64, 67] and image reconstruction in magnetic resonance imaging (MRI) (Fourier inversion) [65, 46]. However, in these papers the authors discard providing any randomness to the generator, instead only giving it the prior. They have thus not fully realized the potential of using GANs for sampling from the posterior in Bayesian inversion.

Fig. 4: The suspected tumor (red) and the reference region (blue) shown in the sample posterior mean image. Right plot shows average contrast differences between the tumor and reference region. The histogram is computed by posterior sampling applied to test data (fig. 2), the yellow curve is from direct estimation, and the true value is the red threshold.

Contrast, the approach based on posterior sampling seems to give a more realistic estimate of the variance.
Regarding sampling from a posterior, conditional generative models [48, 49] have been widely used in the machine learning literature for this purpose. Typical use cases is to sample from a posterior where an image is conditioned on a text, like “the bird is yellow” [32, 20], but also for simple applications in imaging, including image super-resolution and in-painting [44, 52, 51]. These approaches do not consider sampling from the posterior for more elaborate inverse problems that involve a physics driven data likelihood. An approach in this direction is presented in [41] where variational auto-encoders are used to sample from the posterior of possible segmentations (model parameter) given CT images (data).

An entirely different class of methods for exploring the posterior are based on Markov chain Monte Carlo (MCMC) techniques, which have revolutionized mathematical computation and enabled statistical inference within many previously intractable models. Most of the techniques are rooted in solid mathematical theory, but they are limited to cases where the prior model is known in closed form, see surveys in [19, 15, 11]. Furthermore, these MCMC techniques are still computationally unfeasible for large-scale inverse problems, like 3D clinical CT.

A computationally feasible alternative to MCMC for uncertainty quantification is to consider asymptotic characterizations of the posterior. For many inverse problems, it is possible to prove Bernstein–von Mises type of theorems that characterizes the posterior using analytic expressions assuming the prior is asymptotically uninformative [50]. Such characterizations do not hold for finite data, but assuming a Gaussian process model (data likelihood and prior are both Gaussian) allows for using numerical methods for linear inverse problems [55]. Gaussian process models are however still computationally demanding and it can be hard to design appropriate priors, so [23, 24] introduces (conditional) neural processes that incorporate deep neural networks into Gaussian process models for learning more general priors.

Finally, another computationally feasible approach for uncertainty quantification is to approximate Bayesian credible sets for the MAP estimator by solving a convex optimization problem [57, 54]. The approach is however restricted to the MAP estimator and furthermore, it requires access to a handcrafted prior.

7 Conclusions

Bayesian inversion is an elegant framework for recovering model parameters along with uncertainties that applies to a wide range of inverse problems. The
traditional approach requires specifying a prior and, depending on the choice of estimator, also the probability of data. Furthermore, exploring the posterior remains a computational challenge. Hence, despite significant progress in theory and algorithms, Bayesian inversion remains unfeasible for most large scale inverse problems, like those arising in imaging.

This paper addresses all these issues, thereby opening up for the possibility to perform Bayesian inversion on large scale inverse problems. Capitalizing on recent advances in deep learning, we present two approaches for performing Bayesian inversion: Deep Posterior Sampling (section 4.1), which uses a GAN to sample from the posterior, and Deep Direct Estimation (section 4.2) that computes an estimator directly using a deep neural network.

The performance of both approaches is demonstrated in the context of ultra low dose (2% of normal dose) clinical 3D helical CT imaging (section 5). We show how to compute basic Bayesian estimators, like the posterior mean and point-wise standard deviation. We also compute Bayesian credible sets and use this for testing whether a suspected “dark spot” in the liver, which is visible in the posterior mean image, is real. The quality of the posterior mean reconstruction is also quite promising, especially bearing in mind that it is computed from CT data that corresponds to 2% of normal dose.

To the best of our knowledge, this is the first time one can perform such computations on large scale inverse problems in a timely manner, like clinical 3D helical CT image reconstruction. On the other hand, using such a radically different way to perform image reconstruction in clinical practice quickly gets complicated since it must be preceded by clinical trials in the context of image guided decision making. However, there are many advantages that comes with using our proposed approach, which for medical imaging means integrating imaging with clinical decision making while accounting for the uncertainty.

To conclude, the posterior sampling approach allows one to perform Bayesian inversion on large scale inverse problems that goes beyond computing specific estimators, such as the MAP or conditional mean. The framework is not specific to tomography, it applies to essentially any inverse problem assuming access to sufficient amount of “good” supervised training data. Furthermore, the possibility to efficiently sample from the posterior opens up for new ways to integrate decision making with reconstruction.

8 Discussion and Outlook

There are several open research topics related to using GANs as generative models for the posterior in inverse problems.
One natural topic is to have a precise notion of “good” supervised training data. Specifically, it is desirable to estimate the amount of supervised training data necessary for “resolving” the posterior/estimator up to some accuracy. Unfortunately, most of the current theory for Bayesian inference does not apply directly to this setting. Its emphasis is on characterizing the posterior in the asymptotic regime where information content in data increases indefinitely and the prior is asymptotically non-informative, like when a Gaussian prior is used.

Another research topic is to study whether there are theoretical guarantees that ensure the conditional WGAN generator given by (3) converges towards the posterior. In [26] one proves that given infinite capacity of the discriminator, the optimal generator minimizes the Jensen–Shannon divergence w.r.t. the target distribution. For the case with WGAN, [56, Lemma 6] shows that one can learn the posterior in the sense of [60, Definition 3.1], i.e. solving (10) arbitrarily well, given enough training data. But this does not settle the question of what happens with realistic sample and model capacities. This is part of a more general research theme for investigating the theoretical basis for using GANs trained on supervised data to sample from high dimensional probability distributions [9].

Yet another topic relates to including explicit knowledge about the data likelihood, which in contrast to the prior, can be successfully handcrafted for many inverse problems. This is essential for large-scale inverse problems where the amount of supervised training data is little and there are few opportunities for re-training when data the acquisition protocol changes. In this work, this knowledge was implicitly accounted for by our choice to use a FBP reconstruction as the data. While it can be proven that this is in theory sufficient for generating samples from the posterior [4, section 8], [5, 6] clearly shows that working directly from measured data gives better results. We therefore expect further improvements to our results by using a conditional WGANs based on convolutional neural network (CNN) architectures that integrate a handcrafted data likelihood, such as those provided by learned iterative reconstruction.

Finally, our deep direct estimators were very easy to train with no major complications, but training the generative models for posterior sampling is still complicated and involves quite a bit of fine tuning and “tricks”. We hope that future research in generative models will improve upon this situation.
Acknowledgements

The work was supported by the Swedish Foundation of Strategic Research grant AM13-0049, Industrial PhD grant ID14-0055 and by Elekta. The authors also thank Dr. Cynthia McCollough, the Mayo Clinic, and the American Association of Physicists in Medicine, and acknowledge funding from grants EB017095 and EB017185 from the National Institute of Biomedical Imaging and Bioengineering, for providing the data.
Appendices

A The Wasserstein 1-distance

Let $X$ be a measurable separable Banach Space and $\mathcal{P}_X$ the space of probability measures on $X$. The Wasserstein 1-distance $W: \mathcal{P}_X \times \mathcal{P}_X \to \mathbb{R}$ is a metric on $\mathcal{P}_X$ that can be defined as \cite[Definition 6.1]{References}

$$W(p,q) := \inf_{\mu \in \Pi(p,q)} \mathbb{E}_{(x,v) \sim \mu} \left[ \|x - v\|_X \right] \text{ for } p, q \in \mathcal{P}_X. \quad (5)$$

In the above, $\Pi(p,q) \subset \mathcal{P}_X \times X$ denotes the family of joint probability measures on $X \times X$ that have $p$ and $q$ as marginals. Note also that we assume $\mathcal{P}_X$ only contains measures where the Wasserstein distance takes finite values (Wasserstein space), see \cite[Definition 6.4]{References} for the formal definition.

The Wasserstein 1-distance in (5) can be rewritten using the Kantorovich-Rubinstein dual characterization \cite[Remark 6.5 on p. 95]{References}, resulting in

$$W(p,q) = \sup_{D \in \text{Lip}(X)} \left\{ \mathbb{E}_{x \sim q} [D(x)] - \mathbb{E}_{v \sim p} [D(v)] \right\} \text{ for } p, q \in \mathcal{P}_X. \quad (6)$$

Here, $\text{Lip}(X)$ denotes real-valued 1-Lipschitz maps on $X$, i.e.,

$$D \in \text{Lip}(X) \iff |D(x_1) - D(x_2)| \leq \|x_1 - x_2\|_X \text{ for all } x_1, x_2 \in X.$$ 

The above constraint can be hard to enforce in (6) as is, so following \cite[3]{References} we prefer the gradient characterization:

$$D \in \text{Lip}(X) \iff \|\partial D(x)\|_{X^*} \leq 1 \text{ for all } x \in X,$$

where $\partial$ indicates the Fréchet derivative and $X^*$ is the dual space of $X$. In our setting, $X$ is an $L_2$ space, which is a Hilbert space so $X^* = X$ and the Fréchet derivative becomes the (Hilbert space) gradient of $D$.

B Individual Posterior Samples

It is instructive to visually inspect individual random samples of the posterior obtained from the conditional Wasserstein GAN (WGAN) generator.

Generating one such sample is fast, taking approximately 40 ms on a desktop “gaming” PC. Furthermore, as seen in fig. 5, the generated samples look realistic, practically indistinguishable from the ground truth to the untrained
observer. With that said, some anatomical features are clearly misplaced, e.g., there are white "blobs" (blood vessels) in the liver. These are present because the supervised training set contained images from patients that were given contrast (see bottom row in fig. 1), which has influenced the anatomical prior that is learned from the supervised data.

C Theory of Deep Bayesian Inversion

This section contains the theoretical foundations needed for Deep Bayesian Inversion and derivations of the expressions used in the main article.

C.1 Derivation of conditional WGAN

This section provides the mathematical details for deriving (3) from (2). Such a reformulation is well-known, but the derivation given here seems to be novel.

The overall aim with WGAN is to approximate the posterior $y \mapsto \pi(x \mid y)$ that is inaccessible. The approach is to construct a mapping that associates a probability measure on $X$ to each data $y \in Y$. A family of such mappings can be explicitly constructed and trained against supervised training data in order to approximate the posterior. To proceed we need to specify the
statistical setting and our starting point is to introduce a “distance” between two probability measures on $X$:

$$\ell: \mathcal{P}_X \times \mathcal{P}_X \to \mathbb{R}_+.$$  \hspace{1cm} (7)

In the above, $\mathcal{P}_X$ denotes the class of all probability measures on $X$, so $\pi(x \mid y) \in \mathcal{P}_X$ whenever the posterior exists, which holds under fairly general assumptions where both $X$ and $Y$ can be infinite-dimensional separable Banach spaces [19, Theorem 14]. Next, let $\mathfrak{G}$ denote a fixed family of generators, which are mappings

$$\mathcal{G}: Y \to \mathcal{P}_X$$  \hspace{1cm} (8)

that associate each $y \in Y$ to a probability measure on $X$. Note here that $y \mapsto \pi(x \mid y)$ is not necessarily contained in $\mathfrak{G}$. A generator in $\mathfrak{G}$ is an “optimal” approximation of the posterior $y \mapsto \pi(x \mid y)$ if it minimizes the expected $\ell$-distance, i.e., it solves

$$\inf_{\mathcal{G} \in \mathfrak{G}} \mathbb{E}_{y \sim \sigma} \left[ \ell(\mathcal{G}(y), \pi(x \mid y)) \right].$$  \hspace{1cm} (9)

Here, $y \sim \sigma$ is the $Y$-valued random variable generating data.

There are three issues that arise if a solution to (9) is to be used as a proxy for the posterior: (i) Evaluating the objective requires access to the very posterior, which we assumed was inaccessible, (ii) the distribution $\sigma$ of data is almost always unknown, so the expectation cannot be computed, and finally (iii) the computational feasibility requires access to an explicit finite dimensional parametrization for constructing generators in $\mathfrak{G}$ that one searches over in (9).

As we shall see next, choosing (7) as the Wasserstein 1-distance allows us to address the first two issues. With this choice one can re-write the objective in (9) as an expectation over the joint law $(x, y) \sim \mu$, thereby avoiding expressions that explicitly depend on the unknown posterior $y \mapsto \pi(x \mid y)$ and distribution of data $\sigma$. This joint law is also unknown, but it can often be replaced by its empirical counterpart derived from a suitable supervised training data set.

More precisely, choosing the Wasserstein 1-distance $\mathcal{W}: \mathcal{P}_X \times \mathcal{P}_X \to \mathbb{R}_+$ as $\ell$ in (9) yields

$$\inf_{\mathcal{G} \in \mathfrak{G}} \mathbb{E}_{y \sim \sigma} \left[ \mathcal{W}(\mathcal{G}(y), \pi(x \mid y)) \right].$$  \hspace{1cm} (10)

Note here that $y \mapsto \mathcal{W}(\mathcal{G}(y), \pi(x \mid y))$ is assumed to be a measurable real-valued function on $Y$. The Kantorovich-Rubinstein dual characterization in
\( \mathcal{W}(\mathcal{G}(y), \pi(x | y)) = \sup_{D_y \in \text{Lip}(X)} \left\{ \mathbb{E}_{x \sim \pi(x | y)} [D_y(x) - D_y(v)] \right\} \) for \( y \in Y \). (11)

Here, \( \text{Lip}(X) \) denotes the set of real-valued mappings on \( X \) that are 1-Lipschitz. Hence, (10) can be written as

\[
\inf_{\mathcal{G} \in \mathcal{G}} \mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} \left\{ \mathbb{E}_{x \sim \pi(x | y)} [D_y(x) - D_y(v)] \right\} \right].
\] (12)

Next, in this case the supremum commutes with the \( \sigma \)-expectation, i.e.,

\[
\mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} \left\{ \mathbb{E}_{x \sim \pi(x | y)} [D_y(x) - D_y(v)] \right\} \right] = \sup_{D \in \mathcal{D}(X \times Y)} \left\{ \mathbb{E}_{(x,y) \sim \mu} [D(x,y) - D(v,y)] \right\},
\] (13)

where \( \mathcal{D}(X \times Y) \) is the space of measurable real-valued mappings on \( X \times Y \) that are 1-Lipschitz in the \( X \)-variable for every \( y \in Y \). The proof of (13) is given on p. 22 and combining it with (12) gives

\[
\inf_{\mathcal{G} \in \mathcal{G}} \sup_{D \in \mathcal{D}(X \times Y)} \left\{ \mathbb{E}_{(x,y) \sim \mu} [D(x,y) - D(v,y)] \right\}. \] (14)

Note that there are no approximations involved in going from (10) to (14), the derivation is solely based on properties of the Wasserstein 1-distance. Furthermore, the advantage of (14) over (10) is that the latter neither involves the posterior nor \( \sigma \) (the probability measure of data). It does involve the joint law \( (x,y) \sim \mu \), which is of course unknown. On the other hand, if we have access to supervised training data:

\[
(x_1, y_1), \ldots, (x_m, y_m) \in X \times Y \quad \text{are i.i.d. samples of} \quad (x,y) \sim \mu,
\] (15)

then we can replace the joint law \( \mu \) in (14) with its empirical counterpart and the \( \mu \)-expectation is replaced by an averaging over training data.

The final steps concern computational feasibility. We start by considering parameterizations of the generators in \( \mathcal{G} \) that enables one to solve (14) in a computational feasible manner. A key aspect is to evaluate the \( \mathcal{G}(y) \)-expectation for any \( y \in Y \) without impairing upon the ability to approximate
the posterior with elements from $G$. We will assume that each generator $G \in \mathcal{G}$ corresponds to a measurable map $G: Z \times Y \to X$ such that the following holds:

$$v \sim G(y) \iff v = G(z, y) \quad \text{for some Z-valued random variable } z \sim \eta.$$  (16)

In the above, $Z$ is some fixed set and $\eta$ is a “simple” probability measure on $Z$ meaning that there are computationally efficient means for generating samples of $z \sim \eta$. It is then possible to express (14) as

$$\inf_{G \in \mathcal{G}} \left\{ \sup_{D \in \mathcal{D}(X \times Y)} \mathbb{E}_{(x,y) \sim \mu}[D(x,y) - D(G(z,y), y)] \right\}$$  (17)

where $\mathcal{G}$ is the class of $X$-valued measurable maps on $Z \times Y$ that corresponds to $G$ by (16).

The formulation in (17) involves taking the infimum over $\mathcal{G}$ and supremum over $\mathcal{D}$, which is clearly computationally unfeasible. Hence, one option is to consider a parametrization of these spaces using deep neural networks with appropriately chosen architectures:

$$\mathcal{G} := \{G_\theta \}_{\theta \in \Theta} \quad \text{where } G_\theta : Z \times Y \to X$$  (18)

$$\mathcal{D} := \{D_\phi \}_{\phi \in \Phi} \quad \text{where } D_\phi : X \times Y \to \mathbb{R}.$$  (19)

Inserting the above parametrizations into (17) results in

$$\theta^* \in \arg \min_{\theta \in \Theta} \left\{ \sup_{\phi \in \Phi} \mathbb{E}_{(x,y) \sim \mu}[D_\phi(x,y) - D_\phi(G_\theta(z,y), y)] \right\}. \quad (20)$$

Note again that the unknown joint law $\mu$ in (20) is replaced by its empirical counterpart given from the training data in (15).

To summarize, solving the training problem in (20) given training data (15) and the parametrizations in (18) and (19) yields a mapping $G_{\theta^*}: Z \times Y \to X$ that approximates the posterior in the sense that the distribution of $G_{\theta^*}(z, y)$ with $z \sim \eta$ is closest to $\pi(x \mid y)$ in expected Wasserstein 1-distance. Hence, we can sample $z \in Z$ from $z \sim \eta$ and $G_{\theta^*}(z, y) \in X$ will approximate a sample of the conditional random variable $(x \mid y = y) \sim \pi(x \mid y)$. The formulation in (20) is also suitable for stochastic gradient descent (SGD), so computational techniques from deep neural networks can be used for solving the empirical expected minimization problem. We conclude with providing a proof of (13).
Proof of (13)  To simplify the notational burden, define \( f_D : Y \to \mathbb{R} \) as
\[
f_D(y) := \mathbb{E}_{v \sim \mathcal{G}(y)} \left[ D(x) - D(v) \right] \quad \text{for} \ D \in \text{Lip}(X).
\]
Next, \((x, y) \sim \mu\) with \(\mu = \pi(x \mid y) \otimes \sigma\), so by the law of total expectation we can re-write the objective in the right-hand side of (13) as
\[
\mathbb{E}_{(x, y) \sim \mu} \left[ D(x, y) - D(v, y) \right] = \mathbb{E}_{y \sim \sigma} \left[ f_D(\cdot, y)(y) \right].
\]
Hence, proving (13) is equivalent to proving
\[
\mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} f_{D_y}(y) \right] = \sup_{D \in \mathcal{G}(X \times Y)} \mathbb{E}_{y \sim \sigma} \left[ f_D(\cdot, y)(y) \right].  \tag{21}
\]
To prove (21), note first that the claim clearly holds when equality is replaced with \(\geq\) since \(D(\cdot, y) \in \text{Lip}(X)\) for any \(D \in \mathcal{G}(X \times Y)\). It remains to prove that strict inequality in (21) cannot hold. In the following, we use a proof by contradiction approach, so assume strict inequality holds:
\[
\mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} f_{D_y}(y) \right] > \sup_{D \in \mathcal{G}(X \times Y)} \mathbb{E}_{y \sim \sigma} \left[ f_D(\cdot, y)(y) \right]. \tag{22}
\]
From (22), there exists \(\varepsilon > 0\) such that
\[
\mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} f_{D_y}(y) \right] - \varepsilon > \sup_{D \in \mathcal{G}(X \times Y)} \mathbb{E}_{y \sim \sigma} \left[ f_D(\cdot, y)(y) \right]. \tag{23}
\]
Next, for any \(y \in Y\) and \(\varepsilon > 0\), there exists \(\hat{D}_y \in \text{Lip}(X)\) such that
\[
\hat{f}_{D_y}(y) > \sup_{D_y \in \text{Lip}(X)} f_{D_y}(y) - \varepsilon \quad \text{holds for any} \ y \in Y. \tag{24}
\]
Assume next that it is possible to choose \(\hat{D}_y\) so that \((x, y) \mapsto \hat{D}_y(x)\) is measurable on \(X \times Y\). This implies that \((x, y) \mapsto \hat{D}_y(x) \in \mathcal{G}(X \times Y)\) since \(\hat{D}_y \in \text{Lip}(X)\) for all \(y \in Y\). Hence, the \(\sigma\)-expectation of \(\hat{f}_{D_y}(y)\) exists and (24) combined with the monotonicity of the expectation gives
\[
\mathbb{E}_{y \sim \sigma} \left[ \hat{f}_{D_y}(y) \right] > \mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} f_{D_y}(y) - \varepsilon \right] = \mathbb{E}_{y \sim \sigma} \left[ \sup_{D_y \in \text{Lip}(X)} f_{D_y}(y) \right] - \varepsilon.
\]
Insert the above into (23) gives

\[ E_{y \sim \sigma} [f_{\hat{D}}(y)] > \sup_{D \in \mathcal{D}(X \times Y)} E_{y \sim \sigma} [f_D(y)]. \tag{25} \]

Since \((x, y) \mapsto \hat{D}_y(x) \in \mathcal{D}(X \times Y)\), the statement in (25) contradicts the definition of the supremum, i.e., (22) leads to a contradiction implying that (21) is true. This concludes the proof.

### C.2 A novel discriminator for conditional WGAN

A generator trained using the formulation in (20) as is will typically learn to ignore the randomness from \(z \sim \eta\). This can be seen in figs. 6 and 7 that replicate the tests performed in figs. 3 and 5 but with a generator trained using (20) as is. Observe that the inter-sample variance is very low, e.g., the conditional mean image in fig. 6 is still very noisy as compared to corresponding images in fig. 3.

An explanation to this phenomena can be found in statistical learning theory. Note that, regardless of the number of supervised training data points \((x_i, y_i)\), the training data only provides a single \(X\)-sample \(x_i\) of the probability measure \(\pi(x \mid y_i)\), which is the posterior at \(y_i\). Since training data only provides a single sample from \(\pi(x \mid y_i)\), training by (20) will result in a generator that only learns how to generate the corresponding single sample thereby generating the same sample repeatedly (mode collapse) [68].

The importance of addressing mode collapse is clearly illustrated in figs. 6 and 7. One approach to avoid mode collapse is to let the discriminator in (12) see multiple samples from \(\pi(x \mid y_i)\), which leads to the idea of mini-batch discriminators [59, 39]. Such an approach is not possible in Bayes inversion since training data only provides access to a single model parameter \(x\) for each data \(y\). In the following we describe a new conditional mini-batch discriminator that is better at avoiding mode collapse in the Bayesian inversion setting.

**Conditional WGAN discriminator** The idea is to let the discriminator distinguish between unordered pairs in \(X\) containing either the model parameter or random samples generated by the generative model. To formalize this, the generative model is trained using the following generalization of (2):

\[ \inf_{G \in \mathbb{G}} E_{y \sim \sigma} \left[ W\left( G(y) \otimes G(y) \cdot \frac{1}{2} \left( \pi(x \mid y) \otimes G(y) \right) \oplus \frac{1}{2} \left( G(y) \otimes \pi(x \mid y) \right) \right) \right] \tag{26} \]
where ⊕ denotes usual summation of measures. Next, we show that one may train a generative model based on (26) instead of (2). The former lets the discriminator see more than a single sample from the posterior, so the resulting learned generator is much less likely to suffer from mode collapse, see fig. 7 for an empirically confirmation of this.

**Claim 1.** A generative model \( G: Y \to P_X \) solves (26) iff it solves (2).

**Proof.** Let \( y \in Y \) be fixed and consider the objective in (26):

\[
W\left(G(y) \otimes G(y), \frac{1}{2} \left( \pi(x \mid y) \otimes G(y) \right) \oplus \frac{1}{2} \left( G(y) \otimes \pi(x \mid y) \right) \right)
= W\left(\frac{1}{2} (G(y) \otimes G(y), \pi(x \mid y) \otimes G(y)) \oplus \frac{1}{2} (\pi(x \mid y) \otimes G(y)) \right)
\propto W\left(\frac{1}{2} (G(y) \otimes G(y)), \frac{1}{2} (\pi(x \mid y) \otimes \pi(x \mid y)) \right).
\]

The last equality above follows from subtracting the measure \( \frac{1}{2} \left( G(y) \otimes G(y) \right) \) from both arguments in the Wasserstein metric and utilizing its translation invariance (which is easiest to see in the Kantorovich-Rubenstein characterization). Next,

\[
W\left(\frac{1}{2} (G(y) \otimes G(y)), \pi(x \mid y) \right) \propto W\left(G(y), \pi(x \mid y) \right),
\]

so a generative model solves (26) if and only if it solves (2). □

Note that in the proof of claim 1, we implicitly assume the Wasserstein distance can be defined on any pair of positive Radon measures with equal mass. This is a trivial extension of the original definition of the Wasserstein distance, which assumes the domain is a pair of probability measures. It is worth noting that one can define “optimal transportation”-like distances between arbitrary positive Radon measures [17, 18].

To proceed, we need to rewrite the training in (26) so that it becomes more tractable, e.g., by removing the explicit appearance of the unknown posterior and probability measure for data. To do that, we yet again resort to the Kantorovich-Rubenstein duality (6). When applied to (26), it yields

\[
\inf_{G \in \mathbb{G}} \mathbb{E}_{y \sim \sigma} \left[ \sup_{D \in \mathcal{D}} \mathbb{E}_{(x_1, x_2) \sim \rho(y)} \left[ D((x_1, x_2), y) - D((v_1, v_2), y) \right] \right].
\]

(27)

Here, \( \rho(y) := \frac{1}{2} (\pi(x \mid y) \otimes G(y)) \oplus \frac{1}{2} (G(y) \otimes \pi(x \mid y)) \) is a probability measure on \( X \times X \) and \( \mathcal{D} \) are measurable maps \( D: (X \times X) \times Y \to \mathbb{R} \) that are 1-Lipschitz.
w.r.t. its \((X \times X)\)-variable. Next, the same arguments used to rewrite (12) as (14) can also be used to rewrite (27) as

\[
\inf_{\varphi \in \Phi} \left\{ \sup_{D \in \mathcal{D}} \mathbb{E}_{v_1, v_2 \sim \varphi(y)} \left[ \frac{1}{2} \left( D\left((x, v_2), y\right) + D\left((v_1, x), y\right) \right) - D\left((v_1, v_2), y\right) \right] \right\}. \tag{28}
\]

In contrast to (27), the formulation in (28) makes no reference to the posterior \(\pi(x | y)\) nor the probability measure \(\sigma\) for data. Instead, it involves an expectation w.r.t. the joint law \(\mu\), which in a practical setting can be replaced by its empirical counterpart given from supervised training data in (15).

The final step is to introduce parameterizations for the generator and discriminator. The generator is parametrized as in (18), whereas the parametrized family \(\mathcal{D} := \{D_\phi\}_{\phi \in \Phi}\) of discriminators are measurable mappings of the type \(D_\phi : (X \times X) \times Y \to \mathbb{R}\) that are 1-Lipschitz in the \((X \times X)\)-variable. Inserting these parametrizations into (28) results in

\[
(\theta^*, \phi^*) \in \arg \min_{\theta \in \Theta} \left\{ \sup_{\phi \in \Phi} \mathbb{E}_{(x, y) \sim \mu} \left[ \frac{1}{2} \left( D_\phi\left((x, G_\theta(z_2, y)), y\right) + D_\phi\left((G_\theta(z_1, y), x), y\right) \right) - D_\phi\left((G_\theta(z_1, y), G_\theta(z_2, y)), y\right) \right] \right\}. \tag{29}
\]

Note again that the unknown joint law \(\mu\) in (29) is replaced by its empirical counterpart given from the training data in (15).

\section*{C.3 Deep Direct Estimation}

The aim here is to show how an appropriately trained deep neural network can be used for approximating a wide range of non-randomized decision rules (estimators) associated, e.g., with uncertainty quantification. This differs from the posterior sampling approach (section 4.1 and appendix C.1) where such estimators are computed empirically by sampling from a trained WGAN generator.

The idea is to extend the approach in [5, 6] for learning estimators that minimizes Bayes risk so that it applies to a wider class of estimators. Our starting point is a well known proposition from probability theory that characterizes the minimizer of the mean squared error loss.
Fig. 6: Replication of fig. 5 without conditional WGAN discriminator shown using the same intensity window. Observe that there is practically no inter-sample variability due to mode collapse, confirming that the conditional WGAN discriminator is essential for posterior sampling.

**Proposition 2.** Assume that $Y$ be a measurable space, $W$ is a measurable Hilbert space, and $y$ and $w$ are $Y$- and $W$-valued random variables, respectively. Then, the conditional expectation $h^*(y) := \mathbb{E}[w | y = y]$ solves

$$\min_{h: Y \rightarrow W} \mathbb{E}\left[\|h(y) - w\|_W^2\right].$$

The minimization above is taken over all $W$-valued measurable functions on $Y$.

**Proof.** Let $h: Y \rightarrow W$ be any measurable function so

$$\mathbb{E}\left[\|h(y) - w\|_W^2\right] = \mathbb{E}\left[\mathbb{E}\left[\|h(y) - w\|_W^2 | y\right]\right].$$

Next, $W$ is a Hilbert space so we can expand the squared norm:

$$\|h(y) - w\|_W^2 = \|h(y) - \mathbb{E}[w | y] + \mathbb{E}[w | y] - w\|_W^2 =$$

$$= \|h(y) - \mathbb{E}[w | y]\|_W^2 + 2\left\langle h(y) - \mathbb{E}[w | y], \mathbb{E}[w | y] - w \right\rangle_W + \|w - \mathbb{E}[w | y]\|_W^2.$$
Fig. 7: Replication of fig. 3 also showing (right most column) the sample mean and sample point-wise standard deviation (pStd) when the conditional WGAN discriminator is not used. The standard deviation grossly underestimated due to mode collapse.

By the law of total expectation and the linearity of the inner product, we get

$$\mathbb{E}\left[2\left<h(y) - \mathbb{E}[w | y], \mathbb{E}[w | y] - w\right>_W | y\right]$$

$$= 2\left<h(y) - \mathbb{E}[w | y], \mathbb{E}[w | y] - \mathbb{E}[w | y]\right>_W$$

$$= 2\left<h(y) - \mathbb{E}[w | y], 0\right>_W = 0$$

and \(\|w - \mathbb{E}[w | y]\|_W^2\) is independent of \(h(y)\). Combining all of this gives

$$\arg \min_{h: Y \rightarrow W} \mathbb{E}\left[\|h(y) - w\|_W^2\right] = \arg \min_{h: Y \rightarrow W} \mathbb{E}\left[\|h(y) - \mathbb{E}[w | y]\|_W^2\right]$$

where \(h^*(y) = \mathbb{E}[w | y]\) is the solution to the right hand side.

Proposition 2 implies in particular that minimizing Bayes risk with a loss given by the mean squared error amounts to computing the conditional mean. This result does not hold when the loss is the 1-norm, which would give the conditional median instead of the conditional mean. In a finite dimensional setting, proposition 2 holds also when the loss is any functional that is the Bregman distance of a convex functional [10].
In the context of Bayesian inversion, \( x \) and \( y \) are the \( X \)- and \( Y \)-valued random variables generating the model parameter and data, respectively. Proposition 2 is then the starting point for studying the relation between the maximum a posteriori (MAP) and conditional mean estimates [14]. In our setting, if \( h^* : Y \to X \) is the estimator that minimizes Bayes risk using squared loss, then proposition 2 (with \( w := x \)) implies that

\[
\begin{align*}
    h^* & \in \arg \min_{h : Y \to W} \mathbb{E} \left[ \| h(y) - w \|_W^2 \right] \\
    & \implies h^*(y) = \mathbb{E}[x | y = y] \quad \text{for } y \in Y. 
\end{align*}
\]  

(30)

Since neural networks are universal function approximators, training a neural network using the mean squared error as loss yields an approximation of the conditional mean.

By selecting some other regression target \( w \) we can approximate estimators other than the conditional mean. As an example, let us consider the point-wise conditional variance which is defined as

\[
\text{pVar}[x | y = y] := \mathbb{E} \left[ (x - \mathbb{E}[x | y = y])^2 | y = y \right].
\]  

(31)

In the following, we show how the (point-wise) conditional variance can be estimated directly using a neural network trained against supervised data, similar to how we estimate the conditional mean. The key step is to re-write the conditional variance as a minimizer of the expectation of some scalar objective w.r.t. the joint law of \((x, y)\).

**Proposition 3.** Assume that \( Y, X \) are measurable spaces and that \( X \) is a Hilbert space. The point-wise variance is then characterized by

\[
\begin{align*}
    \text{pVar}[x | y = y] & \in \arg \min_{h : Y \to X} \mathbb{E} \left[ \| h(y) - (x - \mathbb{E}[x | y = y])^2 \|_X^2 \right] \\
    \text{where the minimization is taken over all } & X\text{-valued measurable functions on } Y. 
\end{align*}
\]  

(32)

The proof follows by applying proposition 2 with \( w := (x - \mathbb{E}[x | y = y])^2 \), which yields

\[
\begin{align*}
    \arg \min_{h : Y \to X} \mathbb{E} \left[ \| h(y) - (x - \mathbb{E}[x | y = y])^2 \|_X^2 \right] & = \mathbb{E} \left[ (x - \mathbb{E}[x | y = y])^2 | y = \cdot \right].
\end{align*}
\]

In practice we don’t have direct access to samples from \((x - \mathbb{E}[x | y = y])^2\), so this cannot be applied as is since we cannot compute the expectation in (32). However, if there is access to supervised training data as in (15), then
the conditional expectation in (32) can be approximated by a deep neural network trained according to (30), $E[x \mid y = y] \approx T_{\theta}^\dagger(y)$. From this training data one can then generate “new” training data of the form

$$\left((x_i - T_{\theta}^\dagger(y_i))^2, y_i\right) \in X \times Y$$

where $(x_i, y_i) \in X \times Y$ is from (15).

This training data is random samples from a $(X \times Y)$-valued random variable that approximately has required distribution.

Finally, the minimization in (32) can be restricted to $X$-valued measurable functions on $Y$ that are parametrized by another deep neural network architecture $h_\phi : Y \to X$. Hence, the conditional point-wise variance can be estimated as $p\text{Var}[x \mid y = y] \approx h_\phi^\ast(y)$ where $\phi^\ast$ is obtained from solving the following training problems:

$$\theta^\ast \in \arg \min_{\theta} \left\{ E_{(x,y)} \left[ \|x - T_{\theta}^\dagger(y)\|_X^2 \right] \right\}$$

$$\phi^\ast \in \arg \min_{\phi} \left\{ E_{(x,y)} \left[ \|h_\phi(y) - (x - T_{\theta^\ast}^\dagger(y))^2\|_X^2 \right] \right\}.$$

Direct estimation is a sample free method that has several advantages against posterior sampling. First, they are much easier to train, generative adversarial networks (GANs) that are used for posterior sampling are known for being notoriously hard to train whereas learned iterative methods that underly direct estimation can be trained using standard approaches. Next, they are much faster. Evaluating a trained deep neural network for direct estimation requires roughly as much computational power as generating a single sample of the posterior in posterior sampling. Since posterior sampling requires several samples to get sufficient statistics, they will require an order of magnitude more time.

A downside with direct estimation is that a separate neural network has to be constructed and trained for each estimator. This is especially problematic in cases where we need to answer patient-specific questions that are perhaps unknown during training. Another is that direct estimation as introduced here can only be used for estimators that can be re-written as a minimizer of the expectation of some scalar objective w.r.t. the joint law of $(x, y)$. It is well known that conditional distributions can be approximated by Edgeworth expansions that in turn contain such terms [53], so in principle any posterior can be approximated in this manner by a series of direct estimations. However, the computations quickly get complicated and the computational and training related advantages of direct estimation quickly diminishes.
Finally, results and corresponding proofs as stated in this section are not fully rigorous in the function space setting. As an example, proposition 3 would in such a setting involve the theory of higher moments of Banach space valued random variables [34], which quickly involves elaborate measure theory. On the other hand, the proofs are straightforward in finite dimensional spaces.

D Implementation Details

D.1 Training data

Training data is clinical 3D helical computed tomography (CT) scans from the Mayo Clinic Low Dose CT challenge [47]. The data was obtained using a Siemens SOMATOM Definition AS+ scanner and consists of ten abdomen CT scans of patients with predominantly liver and lung cancer obtained at normal dose. The scanner is a 64-slice cone beam helical CT that further enhances longitudinal resolution by a periodic motion of the focal spot in the z-direction (z-flying focal spot acquisition) [22]. The x-ray tube peak voltage (kVp) was 100–120 kV, depending on patient size, the exposure time was 500 ms and the tube current was 230–430 mA, again depending on patient size.

The normal dose reconstructions are obtained by applying a filtered back-projection (FBP)-type of reconstruction scheme, provided by the manufacturer of the scanner, on the full data. To obtain the low dose images, we first subsampled data and then added noise. The original data is acquired using a 3-PI acquisition geometry [13], meaning that the helical pitch is chosen to oversample each integration line by a factor of three. We sub-sampled the data by excluding the “upper” and “lower” pitch, which corresponds to data from 1-PI acquisition geometry. This results in a sub-sampling of 33%. Furthermore, we split each dataset into three independent datasets by using every third angle. This gives a further sub-sampling by 33%, for a total subsampling of ≈10. In addition, we added Poisson noise to the data according to [47] until they corresponded to 2% normal dose scans, i.e. roughly 1000 photons per pixel. While electron noise is significant at these dose levels, we chose not to model it.

Standard FBP was applied to the above ultra low dose data with a Hann filter with cutoff 0.4 and the filter frequency was chosen to maximize the peak signal to noise ratio (PSNR) of the ultra low dose reconstructions. The 2D slice size was set to 512 × 512 pixels with a reconstruction diameter of 370–440 mm (depending on patient size) and a slice-thickness of 3 mm. Note that the FBP reconstruction operator is formally not information conserving when
using a cutoff (information is irreversibly lost), which technically invalidates the claim in section 5.1 that FBP may be used as a pre-processing step without any information loss. However, we did not observed any adverse effects in letting $y$ represent FBP reconstructions rather than CT data.

Finally, in order to (approximately) center the images, they were linearly scaled so that zero corresponds to 0 HU and $-1$ to $-1000$ HU. In total, supervised training data consisted of 6498 pairs of semi-independent 2D images at normal and ultra low dose. To further augment the training data during training, we applied random flips (left-right), rotations ($\pm 10^\circ$), adding pixel-wise dequantization noise distributed according to $U(0,1)$ HU, and a random mean-value offset distributed according to $N(0,10)$ HU.

D.2 Neural networks

For simplicity, all networks are based on a similar convolutional neural network (CNN) architecture that consists of the following three building blocks:

- **Averagepooling.** Mapping an $2n \times 2n$ image to a $n \times n$ image by taking the average over $2 \times 2$ pixel blocks.

- **Pixelshuffle (also “space to depth”) [61].** Mapping a $n \times n$ image with $4c$ channels to a $2n \times 2n$ image with $c$ channels by spatially spreading the channels into a $2 \times 2$ block.

- **Residual blocks [31].** A single residual block consists of applying batch normalization to the input, followed by a nonlinearity, convolution, batch normalization, nonlinearity and finally a convolution. This is added to a $1 \times 1$ convolution of the result of the first batch normalization. Such a block is shown in fig. 9b.

Furthermore, unless otherwise stated, the CNN uses $3 \times 3$ convolutions and leaky ReLU ($\alpha = 0.2$) non-linearities [45].

For the generator $G_\theta: Z \times Y \rightarrow X$, direct mean estimator $T^\dagger_\theta: Y \rightarrow X$, and direct variance estimator $h_\phi: Y \rightarrow X$, we used an architecture similar to U-Net [58] combining down-sampling followed by a residual block until the image is $8 \times 8$ pixels. At this point we performed up-samplings combined with concatenating skip-connections until we reach the original $512 \times 512$ pixel resolution. The network architecture is illustrated in fig. 8. For $T^\dagger_\theta$ and $h_\phi$ the input was simply the data $y$. Regarding the generator, we let the random noise $z$ be white noise on $Z := X$, so $G_\theta: X \times X \rightarrow X$. For the generator
and direct mean estimator we also added an additive skip-connection from $y$
to result [35].

Finally, the discriminator $D_{\phi}$ is parametrized using a similar network ar-
chitecture but stopped at the lowest resolution ($8 \times 8$ pixels) and finished with
two fully connected layers (fig. 9a).

### D.3 Training

First, all training procedures involved applying a small $L_2$ regularization
(weight decay) with constant $10^{-4}$ to complement the expected loss. Fur-
thermore, $\hat{\mu}$ will denote the empirical probability measure derived from the
supervised training data (15) that has undergone data augmentation (ap-
pendix D.1).

**Direct estimation** Training the networks in the direct estimation approach
(appendix C.3) amounts to solving

$$
\theta^* \in \arg \min_{\theta} \left\{ E_{(x,y) \sim \hat{\mu}} \left[ \| x - T_{\theta}^\dagger(y) \|_X^2 \right] + 10^{-4}\|\theta\|^2 \right\}
$$

$$
\phi^* \in \arg \min_{\phi} \left\{ E_{(x,y) \sim \hat{\mu}} \left[ \| h_{\phi}(y) - (x - T_{\theta^*}^\dagger(y))^2 \|_X^2 \right] + 10^{-4}\|\phi\|^2 \right\}.
$$

**Posterior sampling** The WGAN loss with the conditional WGAN discrimi-
nator (appendix C.2) is the objective in (29), i.e., it is given by

$$
L_W(\theta, \phi) := \mathbb{E}_{(x,y) \sim \hat{\mu}} \left[ \frac{1}{2} \left( D_{\phi} \left( (x, G_{\theta}(z_1, y)), y \right) + D_{\phi} \left( (G_{\theta}(z_1, y), x), y \right) \right) - D_{\phi} \left( (G_{\theta}(z_1, y), G_{\theta}(z_2, y)), y \right) \right]
$$

The set-up in (29) indicates that the discriminator should always be fully
trained. Following best practice, instead of minimizing $(\theta, \phi) \mapsto L_W(\theta, \phi)$
jointly, we set-up an intertwined scheme where we take one step to minimize
a generator loss $\theta \mapsto L_G(\theta)$ keeping $\phi$ fixed, then we take five steps to minimize
a discriminator loss $\phi \mapsto L_D(\phi)$ keeping $\theta$ fixed. In the following, we explain
how to construct these generator and discriminator losses.

For training the discriminator, note that $\phi \mapsto L_W(\theta, \phi)$ is invariant w.r.t.
adding an arbitrary constant to the discriminator. This causes the training
We levitate this by adding a small penalization

\[ L_{\text{drift}}(\phi) := \mathbb{E}_{(x,y) \sim \hat{\mu}}[D_{\phi}(x, y)^2]. \]

Next, as in [27], we enforce the 1-Lipschitz condition for the discriminator (see appendix A) by adding the following gradient penalty term:

\[ L_{\text{grad}}(\theta, \phi) := \mathbb{E}_{(x, y) \sim \hat{\mu}} \mathbb{E}_{\epsilon \sim U(0, 1)} \left[ \left\| \Gamma_{\theta, \phi}(x, y, z_1, z_2, \epsilon)|_{X^*} - 1 \right\|^2 \right] \]

where \( \Gamma_{\theta, \phi}: X \times Y \times Z \times Z \times [0, 1] \rightarrow X^* \) is given as

\[
\Gamma_{\theta, \phi}(x, y, z_1, z_2, \epsilon) := \frac{1}{2} \left\{ \partial_1 D_\phi \left( \epsilon (x, G_\theta(z_1, y)) + (1 - \epsilon)(G_\theta(z_1, y), G_\theta(z_2, y)) \right), y \right\}
+ \partial_1 D_\phi \left( \epsilon (G_\theta(z_1, y), x) + (1 - \epsilon)(G_\theta(z_1, y), G_\theta(z_2, y)) \right), y \right\}
\]

with \( \partial_1 D_\phi \) denoting the first order partial (Banach space) derivative w.r.t. the \((X \times X)\)-variable of \( D_\phi : (X \times X) \times Y \rightarrow \mathbb{R} \). Then, the loss \( \phi \mapsto L_D(\phi) \) for training the discriminator (for fixed generator \( \theta \)) becomes

\[ L_D(\phi) := -L_W(\theta, \phi) + 10L_{\text{grad}}(\theta, \phi) + 10^{-3}L_{\text{drift}}(\theta, \phi) + 10^{-4}\|\phi\|^2, \quad (33) \]

where the scalings 10 and \( 10^{-3} \) were chosen according to best practice [27, 39] and not hand-tuned by us.

The loss \( \theta \mapsto L_G(\theta) \) for training the generator (for fixed discriminator \( \phi \)) is

\[ L_G(\theta) := L_W(\theta, \phi) + 10^{-4}\|\theta\|^2. \quad (34) \]

**Optimization for training** We used the same optimization method to train all networks (both for direct and posterior sampling), which was the ADAM optimizer [40] with \( \beta_1 = 0.5, \beta_2 = 0.9 \) and 50,000 training steps (\( \approx 8 \) epochs). For the batch normalization [33], we used decay 0.9 and \( \epsilon = 10^{-5} \). Moreover, we reduced the learning rate following Noisy Linear Cosine Decay [12] with default parameters, starting with a learning rate of \( 2 \cdot 10^{-4} \).

Despite our data-augmentation and regularization, we observed some over-fitting during training, and expect that better results than ours could be obtained with more data.
Fig. 8: Residual U-Net network architecture. “down” indicates that a down-sampling is done before the resblock. “up” indicates that an upsampling and a concatenation is done before the residual block. “no-bn” indicates that batch normalization was not used in that residual block. The input has resolution 512² while the smallest images have size 8².
(a) Discriminator network.

(b) Residual unit used in the network.
All convolutions have the same number of output channels.
E Handcrafted Priors

The samples were generated from Gibbs priors of the form \( e^{-S(x)} \) where the regularization functional \( S: X \rightarrow \mathbb{R} \) is chosen as indicated by the caption text for the images in the top row of fig. 1.

An interesting feature is that many of the samples from the priors shown in fig. 1 appear to be generated by a Gaussian random field prior. This may contradict the conventional wisdom that the choice of prior (regularizer) has a significant impact on the end result. However, a closer consideration shows that this behavior is to be expected from theory. It turns out that several priors, including the total variation (TV)-prior \( S(x) := \|\nabla x\|_1 \), converge weakly a standard Gaussian free field as the discretization becomes finer as shown in [43, Theorem 5.3] for the TV-prior and in [42] for Besov space priors. The differences in the regularized solution provided by the MAP estimator are largely due to a small set of relatively unlikely images. In conclusion, using such priors in Bayesian inversion of large scale inverse problems has very little effect over, e.g., using a Gaussian random field prior.

References


