Regularising Inverse Problems with Generative Machine Learning Models

M. A. G. Duff[1] · **N. D. F. Campbell**[2] · **M. J. Ehrhardt**[1]

Abstract

Deep neural network approaches to inverse imaging problems have produced impressive results in the last few years. In this survey paper, we consider the use of generative models in a variational regularisation approach to inverse problems. The considered regularisers penalise images that are far from the range of a generative model that has learned to produce images similar to a training dataset. We name this family *generative regularisers*. The success of generative regularisers depends on the quality of the generative model and so we propose a set of desired criteria to assess generative models and guide future research. In our numerical experiments, we evaluate three common generative models, autoencoders, variational autoencoders and generative adversarial networks, against our desired criteria. We also test three different generative regularisers on the inverse problems of deblurring, deconvolution, and tomography. We show that restricting solutions of the inverse problem to lie exactly in the range of a generative model can give good results but that allowing small deviations from the range of the generator produces more consistent results. Finally, we discuss future directions and open problems in the field.

Keywords Inverse problems · Generative models · Machine learning · Imaging

1 Introduction

Solving an inverse problem is the process of calculating an unknown quantity, $x \in \mathcal{X}$, from observed, potentially noisy, measurements, $y \in \mathcal{Y}$. In this work, $\mathcal{X}$ and $\mathcal{Y}$ are assumed to be real finite-dimensional vector spaces. The two are related by a forward model, $A : \mathcal{X} \to \mathcal{Y}$, that, for simplicity, is assumed to be linear, giving the equation

$$y = Ax. \tag{1}$$

Inverse problems are nearly always ill-posed: There does not exist a unique solution or small deviations in the data lead to large deviations in the solution. In addition, measurement errors or additional, poorly understood physical processes mean that the observations can also come with additional noise. For example, in the case of additive noise (1) becomes $Ax + \epsilon = y$ for some noise, ϵ, an additional unknown. We generally assume the noise model is known or can be approximated. Addressing the ill-posedness is critical for applications where the solution is used to make decisions, such as in medical imaging. Throughout this paper, we focus on image reconstruction problems, where $x \in \mathcal{X}$ is an image, but there are many other applications.

Generally, ill-posed problems are solved by incorporating some prior information; this is often given in the form of a regulariser in a variational regularisation framework [1–3]. Consider the optimisation problem

$$x^* \in \arg\min_x L_y(Ax) + \lambda R(x). \tag{2}$$

The first term $L_y : \mathcal{Y} \to [0, \infty]$ is a similarity measure that measures the distance between the observed data, y, and reconstructed image, x, under the forward model, A. The choice of L is often linked to the noise model assumed for the data. The second term includes the regulariser, $R : \mathcal{X} \to [0, \infty]$, that is small when some desired property

✉ M. A. G. Duff
M.A.G.Duff@bath.ac.uk

N. D. F. Campbell
N.D.F.campbell@bath.ac.uk

M. J. Ehrhardt
M.J.ehrhardt@bath.ac.uk

1 Department of Mathematical Sciences, University of Bath, Bath BA2 7AY, UK

2 Department of Computer Science, University of Bath, Bath BA2 7AY, UK

of the image is fulfilled. The constant λ is a regularisation parameter that balances between the data similarity term and the regularisation term. As examples, Tikhonov regularisation encourages the reconstruction to be small in the 2-norm, while Total Variation (TV) regularisation [4] allows large gradients (*e.g.*edges) to occur only sparsely in the reconstruction. These hand-built regularisers are better suited to some types of images over others, *e.g.*TV is tailored to piece-wise smooth images. A natural question to ask is: given a set of images, which regulariser would work well? Alternatively, how can we produce regularisers that are tailored to specific data or tasks?

There is a wide body of research into learning regularisers. Approaches include using a regulariser to force reconstructions to be sparse in some learned basis for feasible images [5]. Others have included a network trained for denoising [6–8] or removing artefacts [9–11] in the regulariser term, favouring images that are unchanged by the network. More recently, 'adversarial regularisation' [12] uses a neural network trained to discriminate between desired images and undesired images that contain artefacts. For a recent overview of approaches to using deep learning to solve inverse problems, see for example [13].

In this paper, we consider the case where the regulariser depends on a learned generative model. The assumption is that the plausible reconstructions exhibit local regularities, global symmetries or repeating patterns and so naturally lie on some lower-dimensional manifold, a subset of $\mathcal{X}$. A *generator*, $G : \mathcal{Z} \to \mathcal{X}$, takes points from a *latent space*, $\mathcal{Z}$, where $\dim(\mathcal{Z}) \ll \dim(\mathcal{X})$, and outputs images, parameterising this lower-dimensional manifold. In practice, the generator is taken to be a parameterised function, G_θ, with parameters θ, for example, a neural network trained such that the generated points $G_\theta(z) \in \mathcal{X}$ are similar to some pre-defined training set. In this work, we investigate regularisers [14–17] that penalise values of $x \in \mathcal{X}$ that are far from the range of the generator, G, and call this *generative regularisers*. A popular example [14], revisited in Sect. 3.1.1, limits solutions to those that are exactly in the range of the generator,

$$z^* \in \arg\min_z \|AG(z) - y\|_2^2 + \lambda \|z\|_2^2,$$ (3)
$$x^* = G(z^*).$$

Generative regularisers combine the benefits of both a variational regularisation and a data-driven approach. The variational approach builds on the advancements in model-based inverse problems over the last century, while the data-driven approach will provide more specific information than a hand-crafted regulariser. The method remains flexible as the machine learning element is unsupervised and therefore independent of the forward model and the noise type. In this work, we provide a thorough survey of generative regularisers and then go on to test different generative regularisers, inspired by the literature, on deconvolution, compressed sensing and tomography inverse problems. The success of generative regularisers will depend on the quality of the generator. We propose a set of criteria that would be beneficial for a generative model destined for use in inverse problems and demonstrate possible methods of testing generative models against these criteria. To conclude we identify a number of avenues for future directions and open problems in the field.

2 Generative Models

This section provides background on generators and generative models, focusing in particular on three approaches: Autoencoders, Variational Autoencoders and Generative Adversarial Networks.

2.1 Autoencoder (AE)

An AE has two parts, an encoder and a decoder. The encoder encodes an image in some latent space and the decoder takes a point in this latent space and decodes it, outputting an image. The lower-dimensional latent space forces the network to learn representations of the input with reduced dimensionality. For reference see *e.g.* chapter 14 of [18]. Denote the encoder $E_\psi : \mathcal{X} \to \mathcal{Z}$ and the decoder $G_\theta : \mathcal{Z} \to \mathcal{X}$, neural networks with parameters ψ and θ. The networks are trained by minimising a reconstruction loss

$$\mathbb{E}_x \left\| x - G_\theta(E_\psi(x)) \right\|_2^2.$$ (4)

The expectation is taken empirically over the training dataset. Post-training, the decoder can be used as a generator. With no structure imposed on the latent space, generating from random points in the latent space may not lead to outputs similar to the training set. Furthermore, points close together in the latent space may not lead to similar generated images. Nevertheless, this method of training is simple and has recently been used in learned singular valued decomposition and for applications in sparse view CT [19, 20].

2.2 Probabilistic Models

In order to add greater structure and meaning to the latent space and to discourage unrealistic outputs from the generator, we consider a probabilistic approach. Let P^* be the probability distribution of desired solutions to the inverse problem. We consider a prior distribution, $P_\mathcal{Z}$, and push it through the generator, G, to give a generated distribution P_G

on $\mathcal{X}$. Sampling from the prior and then applying the generator allows samples to be taken from P_G. The generator, G, is chosen to minimise a distance between P_G and P^*.

2.3 Generative Adversarial Network (GAN)

The choice of Wasserstein distance between P_G and P^* and an application of the Kantorovich-Rubinstein duality leads to the Wasserstein GAN [21], a popular generative model. Following the derivation given in [21, 22], the task of minimising the Wasserstein distance becomes

$$\min_\theta \max_\psi \ \mathbb{E}_{x \sim P^*} D_\psi(x) - \mathbb{E}_{z \sim P_\mathcal{Z}} D_\psi(G_\theta(z)). \tag{5}$$

The *generator* $G_\theta : \mathcal{Z} \to \mathcal{X}$ is as before, and we have introduced a *discriminator* $D_\psi : \mathcal{X} \to \mathbb{R}$ which must be 1-Lipschitz, enforced by an additional term added to (5) [22].

In the game-theoretic interpretation of the GAN [23], a generative model competes with a discriminative model. The discriminator aims to accurately identify real images, maximising $\mathbb{E}_{x \sim P^*} D_\psi(x)$, from generated images, minimising $\mathbb{E}_{z \sim P_\mathcal{Z}} D_\psi(G_\theta(z))$. The generator tries to force the discriminator to label generated images as real, maximising $\mathbb{E}_{z \sim P_\mathcal{Z}} D_\psi(G_\theta(z))$.

For the numerical results in this paper, we choose to use a Wasserstein GAN (5) as it is often more robust to a range of network designs and there is less evidence of mode collapse, when the generator learns just part of the target distribution, compared to the 'vanilla' GAN [21].

2.4 Variational Autoencoder (VAE)

For another choice of distance between P_G and P^*, take the Kullback–Leibler (KL) divergence $(d_{KL}(P^* \| P_G) = \int_\mathcal{X} \log \frac{dP^*}{dP_G} dP^*)$ which leads to a VAE [24]. Following the derivation in [24, 25], the VAE loss function can be written as

$$\mathbb{E}_{x \sim P^*} \left(\mathbb{E}_{z \sim \mathcal{N}_{x,\psi}} \left[\frac{\|x - G_\theta(z)\|_2^2}{2\rho^2} \right] + d_{KL} \left(\mathcal{N}_{x,\psi} \| P_\mathcal{Z} \right) \right) \tag{6}$$

where $\mathcal{N}_{x,\psi} := \mathcal{N}(\mu_\psi(x), \mathrm{diag}(\sigma_\psi^2(x)))$ and $\mu_\psi, \sigma_\psi^2 : \mathcal{X} \to \mathcal{Z}$ are the *encoder mean* and *encoder variance*, neural networks with parameters ψ. The constant, ρ, is a hyperparameter chosen to reflect the 'noise level' in the training dataset. The KL divergence is calculated over the latent space, $\mathcal{Z}$.

We can interpret the two terms in (6) as a data fit and a regulariser term, respectively. In the first term, the reconstruction and the original image are compared. Encoding to a distribution, $\mathcal{N}_{x,\psi}$, enforces that points close to each other in the latent space should produce similar images. In the second

term, the KL divergence encourages the encoded distributions to be close to the prior, $P_\mathcal{Z}$. The prior is usually taken to be the standard normal distribution. The balance between the two terms is determined by the noise level ρ.

2.5 Other Generative Models

Generative modelling is a fast-growing field and there are other examples of generative models. Autoregressive models [26] generate individual pixels based on a probability distribution conditioned on previously generated pixels. Normalising flows and, more generally, invertible neural networks [27], map between a latent space and an image space of the same dimension and are designed to be bijective, with a tractable Jacobian. They can provide a generated distribution with tractable density. A recent invertible neural network example is a GLOW network [28] which has been used in regularisation of the form $R_G(x) = \|G^{-1}(x)\|_2^2$ [29]. Score-based generative models, learn to approximate ∇p^*, where p^* is a probability density over the desired image distribution, P^*, and can then be used to sample from P^* using Langevin dynamics [30]. They have also been used recently as priors for inverse problems, allowing the approximate posterior to be sampled using Monte Carlo methods [31, 32]. We will consider desired properties of generative models in more detail in Sect. 4.

In this work, we focus on AEs, VAEs and GANs because they satisfy the low-dimensional manifold assumption and we hope that the lower dimensional manifold will have a regularising effect. One concern, especially for use of invertible neural networks for inverse problems, is whether they might add more instabilities to the problem, rather than mitigate the original ill-posedness. There has been some work looking at the instabilities of invertible neural networks [33]. This is a possible avenue for future research. Finally, is also worth noting that, in this work, the training and application of the generative model are separate, so that any function, G_θ, can be included.

3 Generative Regularisers for Inverse Problems

In this section, we bring together current approaches in the literature that penalise solutions of an inverse problem that are far from the range of the generator, G. We consider variational regularisation (2) and regularisers of the form

$$R_G(x) = \min_{z \in \mathcal{Z}} F(G(z) - x) + R_\mathcal{Z}(z) \tag{7}$$

where $F : \mathcal{X} \to [0, \infty]$ and $R_{\mathcal{Z}} : \mathcal{Z} \to [0, \infty]$. We consider choices for F, which we call the *generator fidelity*.

3.1 Choices of Generator Fidelity

3.1.1 Restricting to the Range

The characteristic function of an arbitrary set $\mathcal{C}$ is defined as

$$
\iota_{\mathcal{C}}(t) = \begin{cases} 0 & \text{for } t \in \mathcal{C} \\ \infty & \text{for } t \notin \mathcal{C} \end{cases}.
$$

Taking $F(x) = \iota_{\{0\}}(x)$ and $R_{\mathcal{Z}}(z) = \|z\|_2^2$ in (7) gives (3) and describes searching over the latent space for the encoding that best fits the data. Their choice $R_{\mathcal{Z}}(z)$ reflects the Gaussian prior placed on the latent space. Bora et al. [14] first proposed this strategy, applying it to compressed sensing problems. There are a number of interesting applications using this method, such as denoising [17], semantic manipulation [34], seismic waveform inversion [35], light field reconstruction [36], blind deconvolution [37] and phase retrieval [38]. Bora et al. [14] assume the existence of an optimisation scheme that can minimise (3) with small error and from this probabilistically bound the reconstruction error. However, the non-convexity introduced by the generator makes any theoretical guarantees on the optimisation extremely difficult. Assuming the forward operator is a Gaussian matrix (the generator weights have independent and identically distributed Gaussian entries) and the layers of the generator are sufficiently expansive in size, there exist theoretical results on the success of gradient descent for optimising (3) [39–41].

This formulation can also be optimised by projected gradient descent [42, 43]:

$$
\begin{aligned}
w_{t+1} &= x_t - \eta A^T (A x_t - y) \\
z_{t+1} &= \arg\min_z \|w_{t+1} - G(z)\|_2 \\
x_{t+1} &= G(z_{t+1}).
\end{aligned}
\tag{8}
$$

With analogies to the restricted isometry property in compressed sensing [44], Shah and Hegde [42] introduce the *Set Restricted Eigenvalue Condition (S-REC)*. If the S-REC holds, then the operator A preserves the uniqueness of signals in the range of G. Theoretical work considers the case where A is a random Gaussian matrix, and shows, under some assumptions, it satisfies the S-REC with high probability. In addition, if the generator is an untrained network, then the projected gradient descent approach with sufficiently small step size converges to x^*, where $Ax^* = y$ [42, 43, 45].

3.1.2 Relaxing Constraints

Returning to Bora et al. [14], the authors note that as they increase the number of compressed sensing measurements, the quality of the reconstruction levels off rather than continuing to improve. They hypothesise that this occurs when the ground truth is not in the range of the generator. One could consider relaxing the constraint that the solution is in the range of the generator, for example, setting $F(x) = \|x\|_2^2$ allows for small deviations from the range of the generator. One could also encourage the deviations to be sparse, for example by taking $F(x) = \|x\|_1$ [15, 46]. Some theoretical considerations for this softly constrained approach are given in [11]. This approach is similar to the approaches of [9, 10] where they take $G \circ E : \mathcal{X} \to \mathcal{X}$ an encoder–decoder network and define $R_G(x) = \|x - G(E(x))\|_2^2$. The idea is that this regulariser approximates the distance between x and the ideal data manifold. Less explicitly, there are a number of approaches that extend the range of the original generator, through optimisation of intermediate layers of the network [47–49] or tweaking the generative model training in response to observed data [50, 51].

3.2 Additional Regularisation

Additional regularisation on $\mathcal{Z}$ is given by $R_{\mathcal{Z}}$ in (7). The most common choice is $R_{\mathcal{Z}}(z) = \|z\|_2^2$ [14, 37] but there are other possibilities, for example $R_{\mathcal{Z}}(z) = \iota_{[-1,1]^d}(z)$ [17], where $d = \dim \mathcal{Z}$. Often, the regularisation matches the prior on the latent space used in generator training. Menon et al. [47] discuss that $R_{\mathcal{Z}}(z) = \|z\|_2^2$ forces latent vectors towards the origin. However, most of the mass of the d-dimensional standard normal the prior on their latent space is located near the surface of a sphere of radius d. Instead, they use a uniform prior on $d\mathcal{S}^{d-1}$. This idea has also been explored for interpolations in the latent space [52]. In addition, the prior on the latent space may not be a good model for the post-training subset of z that maps to feasible images. For a VAE there may be areas of the latent space that the generator has not seen in training and for a GAN, there could be mode collapse. A few recent papers consider how to find the post-training latent space distribution [53, 54].

Other regularisation choices could be based on features of the image, $x = G(z)$. For example VanVeen et al. [55] use $R_{\mathcal{Z}}(z) = \text{TV}(G(z))$. For a GAN generator, it is possible to take the regularisation term to be the same as the generator loss $R_{\mathcal{Z}}(z) = \log(1 - D(G(z)))$. This regulariser utilises the discriminator, D, which has been trained to differentiate generated from real data. Examples include inpainting [56, 57] and reconstruction from an unknown forward model [58].

3.3 Other Approaches

There are a number of ideas that are linked to earlier discussions in this section but we will not cover them in detail. A major benefit of (2) is the flexibility to changes in the forward model. We have therefore ignored conditional generative models [59–65] and those that train with a specific forward model in mind [9, 66, 67]. We also exclude work that uses an untrained neural network, for example, Deep Image Priors [55, 68, 69] or [16].

4 Generative Model Evaluation

Typically the aim of a generator has been to produce high fidelity images. However, the success of (7) relies not just on the ability of the generator to produce a few good images but to be able to produce every possible feasible image. In this section, we discuss desired properties for a generator trained for use in inverse problems and numerically explore methods to test these properties.

4.1 Desired Properties

To evaluate a generative model, in the context of inverse problems, we consider two overall aims which we will go on to further decompose:

A Samples from the generator are similar to those from the target distribution.

B Given a forward model and an observation, the image in the range of the generator that best fits the observation can be recovered using descent methods.

We split aim A into a set of properties:

A1 The generator should be able to produce every possible image in the target distribution. That is, for all $x \in \mathcal{X}$ such that x is similar to images in the training dataset, there exists $z \in \mathcal{Z}$ such that $G(z) = x$.

A2 The generator should not be able to produce images far from the target distribution. That is, for all $x \in \mathcal{X}$ such that x is not similar to images in the training dataset, then there does not exist $z \in \mathcal{Z}$ such that $G(z) = x$.

A1 includes that the generator should be robust to mode collapse and that the model should not trivially over-fit to the training data.

In the probabilistic case, with a prior over the latent space, property A becomes:

A That samples from the latent space, when mapped through the generator, will produce samples that

approximate a target distribution. We should have that $d(P^*, P_G)$ is small for some distance measure d.

We also note that in the probabilistic case, A1 and A2 are not independent. By assigning probability mass to parts of the image space close to the target distribution, it is less likely that images far from the target distribution can be generated. In the probabilistic case, a third property is added:

A3 The generator should map high-probability vectors in the latent space distribution to high-probability images in the target distribution.

It is possible that A1 and A2 are satisfied but not A3. Note that these properties may not be possible to achieve for a given dataset.

We define two properties for property B, these are

B1 The generator should be smooth with respect to the latent space, $\mathcal{Z}$.

B2 The area of the latent space, $\mathcal{Z}$, that corresponds to images similar to those in the training set should be known.

Property B1 ensures that gradient-based optimisation methods can be used. Continuity is also desirable: we wish that, in some way, points close together in the latent space should produce similar images. B2 considers that we need to have a distribution on or subset of $\mathcal{Z}$ to sample from in order to use the generator to sample images. This distribution may not necessarily be equal to any priors on the latent space used during training. We recognise that B1 and B2 are perhaps vague, and are potentially not sufficient for property B. It is an area for future work to consider making these statements precise enough to support theoretical work.

4.2 Generative Model Evaluation Methods

There are a wide range of existing generative model evaluation methods [70], focused mostly on property A. We assume the availability of some test data drawn from the same distribution as the training data and unseen by the generative model. The *average log likelihood* [23] of test data under the generated distribution is a natural objective to maximise. There is evidence, however, that the likelihood is generally unrelated to image quality and is difficult to approximate in higher dimensions [71]. To calculate a distance between generated and desired distributions, one possibility is the earth movers distance (EMD) [72], a discretised version of the Wasserstein distance. One could also encode the generated and unseen data in a lower-dimensional space before taking distance calculations, for example by taking the outputs of one layer of any neural network trained for classification [73, 74]. A model that overfits to the training data would

perform perfectly in these distance measures. Also, the low-dimensional representation used for the evaluation is likely to have the same inherent problems and drawbacks as the embedding learnt by the generative model. Similarly, a number of tests train an additional, separate, neural network discriminator to distinguish between test data and generated data [21, 75]. Failure to classify the two is a success of the generative model. For testing a GAN, the new discriminator is unlikely to be able to pick up failures that the original discriminator, used in training, missed. Finally, Arora et al. [76] estimate the size of the support of the generated distribution. A low support size would suggest mode collapse. Their technique depends on manually finding duplicate generated images which can be time-consuming and require expert knowledge.

Property B is less explored in the literature. One approach is to directly attempt to reconstruct test data by finding a latent space vector that when pushed through the generator, matches the data. With these found latent vectors, analysing their locations could check property B2. To test the smoothness of the generator with respect to the latent space, property B1, many previous papers, including the original GAN and VAE papers [23, 24], interpolate through the latent space, checking for smooth transitions in the generated images.

4.3 Numerical Experiments

In this section, we evaluate AE, VAE and GAN models against the desired properties given in Sect. 4.1. We consider experiments on two datasets. Firstly, a custom-made `Shapes` dataset with 60,000 training and 10,000 test 56×56 grey-scale images. Each image consists of a black background with a grey circle and rectangle of constant colour. The radius of the circle; the height and width of the rectangle; and the locations of the two shapes are sampled uniformly with ranges chosen such that the shapes do not overlap. This dataset is similar to the one used in [60]. Secondly, the `MNIST` dataset [77] consists of 28×28 grey-scale images of handwritten digits with a training set of 60,000 samples and a test set of 10,000 samples. For examples of both datasets, see the ground truth images in Fig. 2.

Architecture details are given in the appendix. We chose to use the same generator network for all three models, for comparison. Architecture choices were guided by [22, 24, 78]. All models have gone through a similar amount of optimisation of hyperparameters, including the noise level ρ in the VAE decoder (6); the latent dimension; the number of layers; choice of convolution kernel size; drop out probability; leaky ReLU coefficient and learning rate. In order to select hyperparameters we manually inspected generated images. Models were built and trained using Tensorflow [79] in Python and made use of the Balena High-Performance Computing Service at the University of Bath. The models

were trained using a single Dell PowerEdge C8220X node, with two Intel E5-2650 v2 CPUs, 64 GB DDR3-1866 MHz Memory and an Nvidia K20X GPU, 6 GB memory. The `MNIST` and `Shapes` VAE models take approximately 25 and 45 min to train, respectively.

There has been expansive research in generative models over the past few years including a variety of extensions or adaptations to VAEs and GANs. The exploration in this section is not meant to be exhaustive but instead to both illustrate experiments assessing the desired criteria that could be repeated on other datasets and also to demonstrate some general conclusions about the basic class of AEs, VAEs and GANs applied to these fairly simple but illustrative datasets.

4.3.1 Reconstructing a Test Dataset

Property A1 asks that the generator is able to produce every image in the target distribution. Gradient descent with backtracking line search (Algorithm 1, in the appendix) is used to approximate

$$z^*(x) \in \arg\min_z \|G(z) - x\|_2^2, \tag{9}$$

for each $x \in \mathcal{X}_{\text{test}}$, an unseen test dataset. For the AE and VAE, the algorithm is initialised at the (mean) encoding of the test image, $E_\psi(x)$ and $\mu_\psi(x)$, respectively. For the GAN, we take 4 different initialisations, drawn from a standard normal distribution, and take the best result. We find empirically, especially for the GAN, that different initialisations lead to different solutions.

Figure 1 shows $\|G(z^*(x)) - x\|_2/\|x\|_2$, the normalised root mean squared error (NRMSE) for reconstructions on `Shapes` and `MNIST` for the three different generator models. We see that, for both datasets, the AE and VAE have almost identical reconstruction results and the GAN results are comparatively worse. For the `Shapes` dataset the difference in results between the three generative models is less stark. In addition, NRMSE values are given for three different latent dimensions to show that the results are not sensitive to small changes in the latent dimension. Latent dimensions of 8 and 10 for `MNIST` and `Shapes`, respectively, are used in the rest of this paper. Figure 2 also shows reconstruction examples providing context to the results in Fig. 1. Numerical values on the image use the Peak-Signal-to-Noise-Ratio (PSNR, see definition 3.5 in [80]). The non-circular objects in the GAN results for `Shapes` could be a failure of the discriminator to detect circles.

4.3.2 Distance Between P_G and P^*

To investigate property A3, the EMD [72] is calculated between empirical observations of the generated and the data
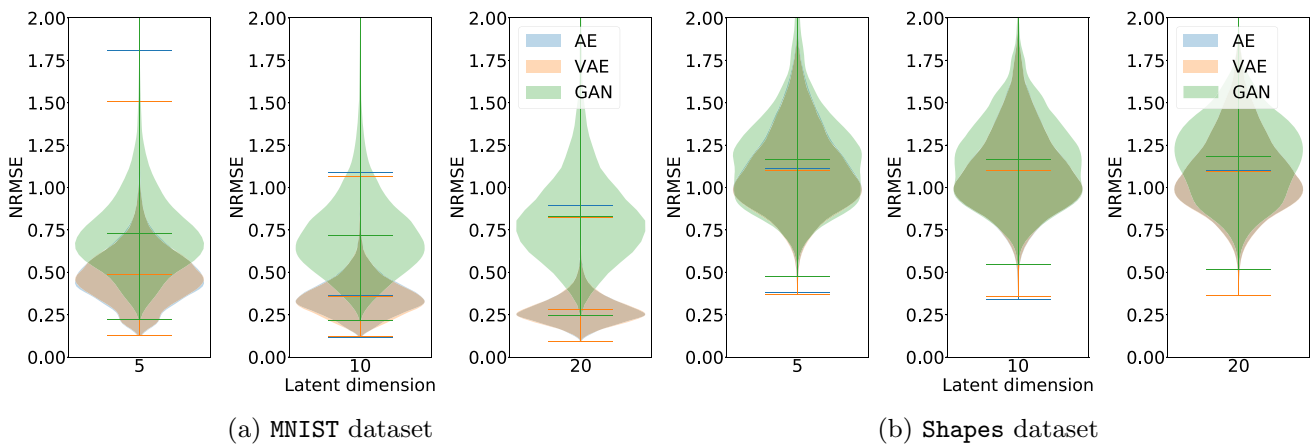
Fig. 1 NRMSE between values of G ($\arg\min_z \|G(z) - x\|_2$) and x and plotted as a histogram for all $x \in \mathcal{X}_{\text{test}}$. The horizontal lines show the median and range and the shaded area is a histogram. Note the brown colour is the result of the overlapping orange (VAE) and blue (AE) (Color figure online)
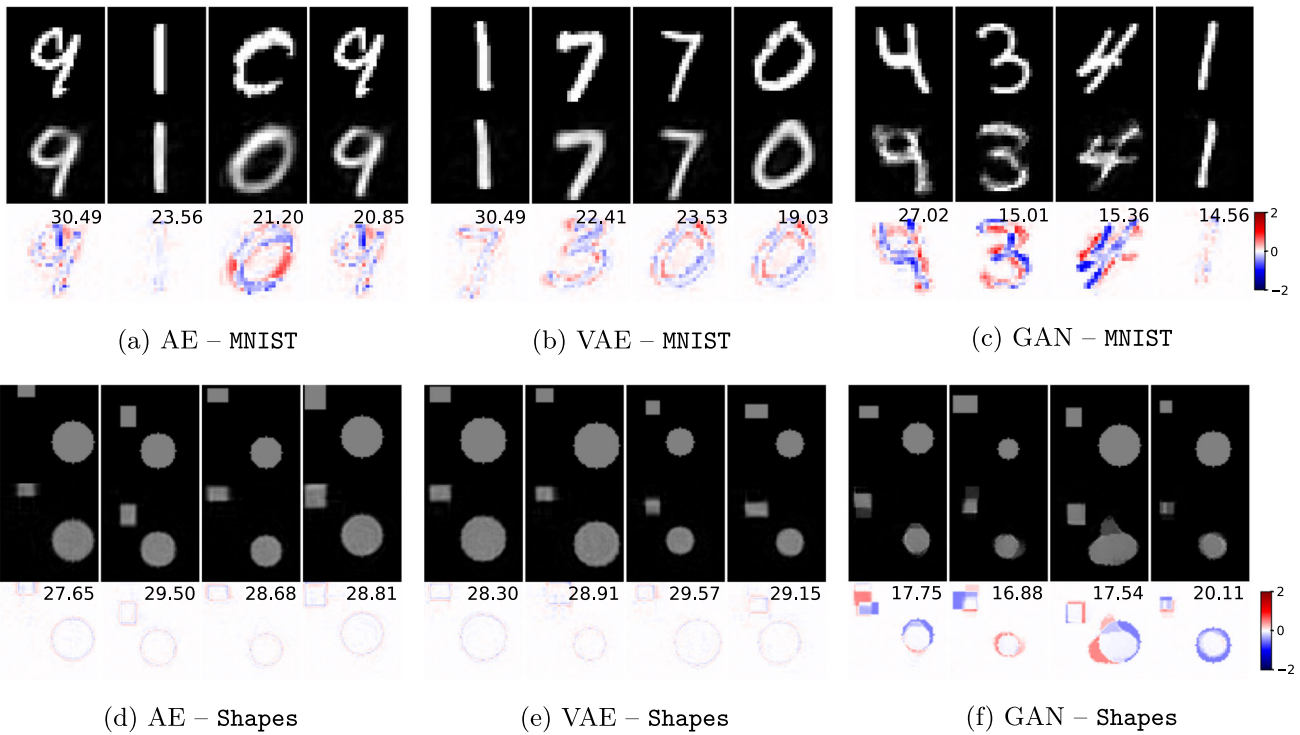

Fig. 2 Example reconstructions for the MNIST and Shapes dataset with eight ten-dimensional generative models, respectively. In each sub-figure, the top row shows the ground truth, the second row the reconstruction and the third row the difference between the two (Color figure online)

distributions P_G and P^*. For the sets of test and generated images, $\mathcal{X}_{\text{test}} = \{x_1, \ldots, x_N\}$ and $\{G(z_1), \ldots, G(z_N) : z_i \sim \mathcal{N}(0, I)\}$, the EMD between their empirical distributions is defined as

$$\min_f \sum_{i,j=1}^{N} f_{i,j} \|x_i - G(z_j)\|_2^2 \qquad (10)$$

where $0 \leq f_{i,j} \leq 1$, $\sum_{i=1}^{N} f_{i,j} = 1$ and $\sum_{j=1}^{N} f_{i,j} = 1$.

The EMD is calculated using the Python Optimal Transport Library [81] with $N = 10,000$, the full test set. The results are given in Fig. 3. In both the MNIST and Shapes examples, the VAE has a lower EMD across the latent dimensions. The AE is added to this plot for comparison purposes but, as there is no prior on the latent space, $z_i \sim \mathcal{N}(0, I)$ may not be a suitable choice to sample from.
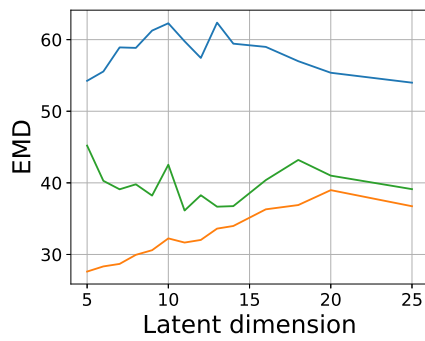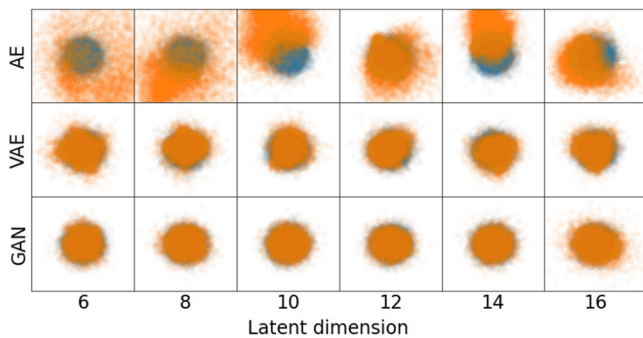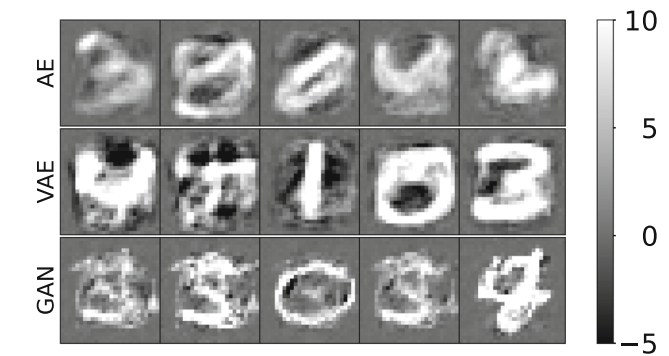
(a) MNIST
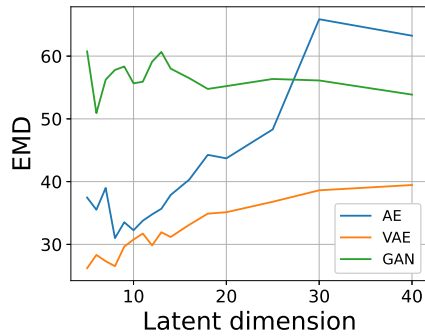
(b) Shapes

Fig. 3 EMD between the test dataset and samples from a trained generator (Color figure online)

(a) MNIST

(b) Shapes

Fig. 5 Images generated far from the high-probability region of the prior distribution

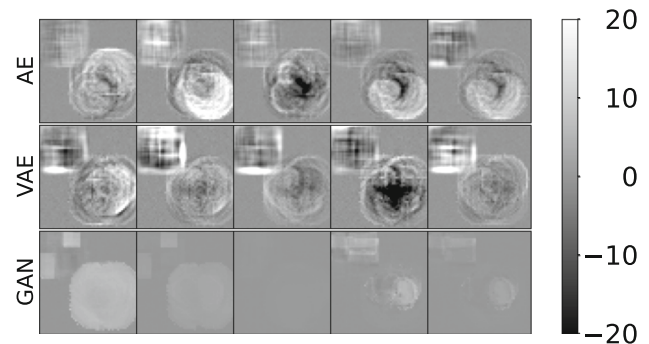### 4.3.3 Latent Space Visualisations

Property B2 requires that the area of the latent space that maps to feasible images is known. There is no prior on the latent space enforced for AEs and a $\mathcal{N}(0, I)$ prior is imposed for VAEs and GANs. In Fig. 4, gradient descent with backtracking (Algorithm 1 in the appendix) is used to approximate (9), finding a latent vector $z^*(x)$ for each $x \in \mathcal{X}_{\text{test}}$. For comparison, the values $z^*(x)$ for the test set and 10,000 vectors drawn from a standard normal distribution are randomly projected into 2 dimensions. The encodings in the latent space match

the prior $\mathcal{N}(0, I)$ for VAEs and GANs. For AEs, there are examples in lower latent dimensions, where the area covered by the encodings does not match a standard normal distribution.

4.3.4 Generating Far from the Latent Distribution

A known latent space gives known areas to sample from to produce new images. Figure 5 shows image examples gen-

(a) MNIST

(b) Shapes

Fig. 4 Comparisons of the latent space encodings of a test dataset with a standard normal distribution by projecting the vectors into 2 dimensions. Encodings of the test dataset are in orange and the standard normal vectors are in blue (Color figure online)

erated far from a standard normal distribution. The images are not recognisable as similar to the training datasets. This emphasises the importance of property B2, that the area of the latent space that corresponds to images similar to those in the training set should be known.

4.3.5 Latent Space Interpolations

We consider interpolating between points in the latent space, testing property B1. We hope to see smooth transitions between interpolated images, and that generated images are similar to those seen in training. We take three images from the test data, x_1, x_2 and x_3, find z_1, z_2 and z_3, their encodings in the latent space, using (9) and then plot interpolations $G(z_1 + \alpha_1(z_2 - z_1) + \alpha_2(z_3 - z_1))$ for $\alpha_1, \alpha_2 \in [0, 1]$. Figure 6 shows one example for each model and dataset for $\alpha_1, \alpha_2 = \{0, 0.25, 0.5, 0.75, 1\}$. For the AE and VAE, you see transitions that are smooth but blurry. The GAN images appear sharper but some outputs are not similar to training data examples, for example, in Fig. 6f there are a set of images that contain no rectangle. These images could be evidence of a discriminator failure: The discriminator has not yet learnt that these images are not similar to the training set.

4.3.6 Discussion

As expected, none of the three generator models, AE, VAE and GAN, fulfill properties A and B fully. For A, the GAN does poorly in the reconstruction results of Figs. 1 and 2. The lack of an encoder makes this more challenging. There is evidence of mode collapse, where parts of the training data are not well reconstructed and discriminator failure, where the images produced are not realistic, see Figs. 2 and 6. The VAE does consistently better, demonstrated by the lower EMD between generated and test data in Fig. 3. The lack of prior on the AE, and thus a known area of the latent space to sample from, is a problem. Figure 5 demonstrates that sampling from the wrong area of the latent space gives poor results.

Pulling apart the cause of a failure to recover an image is difficult. It could be that the image is not in the range of the generator, a failure of property A, or that the image is in the range of the generator but the image cannot be recovered using descent methods, a failure of property B. For property B1, the mathematical properties of continuity or differentiability of a network, depending on the architecture. The interpolations in Fig. 6 show some evidence of large jumps between images in the GAN cases, but in general, the interpolations are reasonable. For both the GAN and the VAE, in Fig. 4, the encodings of the test images in the latent space seem to match the prior, property B2.

Considering the objective functions for the AE, VAE and GAN in (4), (6) and (5), respectively, much of these results are expected. There is no requirement in the GAN objective to cover the range of the training distribution, allowing for mode collapse, or that points close in the latent space map to similar images. In contrast, the expectation over the training set and expectation over the encoded distribution encourage these properties for the VAE. We might have expected some visually better images for the GAN but note that they are often difficult to train [78]. The AE results are more surprising, giving similar results to the VAE in many experiments. This is potentially due to the simplicity of the test datasets but may also point towards evidence of implicit regularisation on the AE from the architecture choices and training algorithms.

5 Numerical Results for Inverse Problems

In this section, we apply the AE, VAE and GAN models, evaluated in the previous section, on three inverse problems. Firstly *tomography*, the X-ray transform [82] with a parallel beam geometry. Secondly, *deconvolution* with a 5×5 Gaussian kernel. Lastly, *compressed sensing* where $y = Ax$ is an under-determined linear system where A is an $\mathbb{R}^{m \times n}$ Gaussian random matrix, $x \in \mathbb{R}^n$ is a vectorised image and $n \gg m$, see for example [83]. In each case, zero-mean Gaussian noise with standard deviation σ is added to the data. The forward operators were implemented using the operator discretisation library (ODL) [84] in Python, accessing scikit–learn [85] for the tomography back-end.

We consider variational regularisation methods in the form of (2) and (7) with $L_y(Ax) = \|Ax - y\|_2^2$. To match the literature themes, we compare three different methods: hard, $F(u) = \iota_{\{0\}}(u)$ and $R_{\mathcal{Z}}(z) = \|z\|_2^2$; relaxed, $F(u) = \|u\|_2^2$ and $R_{\mathcal{Z}}(z) = \mu\|z\|_2^2$; and sparse, $F(u) = \|u\|_1$ and $R_{\mathcal{Z}}(z) = \mu\|z\|_2^2$, where μ is an additional regularisation parameter. We compare with regularisers independent of the generator: Tikhonov regularisation, $R_G(x) = \|x\|_2^2$, for the convolution and tomography examples and TV regularisation [4], for the compressed sensing example.

The optimisation algorithms are given in the appendix. Hard and Tikhonov are optimised using gradient descent with backtracking line search, Algorithm 1. TV regularisation is implemented using the Primal-Dual Hybrid Gradient method [86]. For relaxed, alternating gradient descent with backtracking is used, see Algorithm 2. Finally, for sparse, the 1-norm is not smooth, and so Proximal Alternating Linearised Minimisation (PALM) [87] with backtracking, Algorithm 3, is used to optimise the equivalent formulation $\min_{u \in \mathcal{X}, z \in \mathcal{Z}} \|A(G(z) + u) - y\|_2^2 + \lambda \left(\|u\|_1 + \mu\|z\|_2^2 \right)$. In all cases, initial values are chosen from a standard normal distribution.
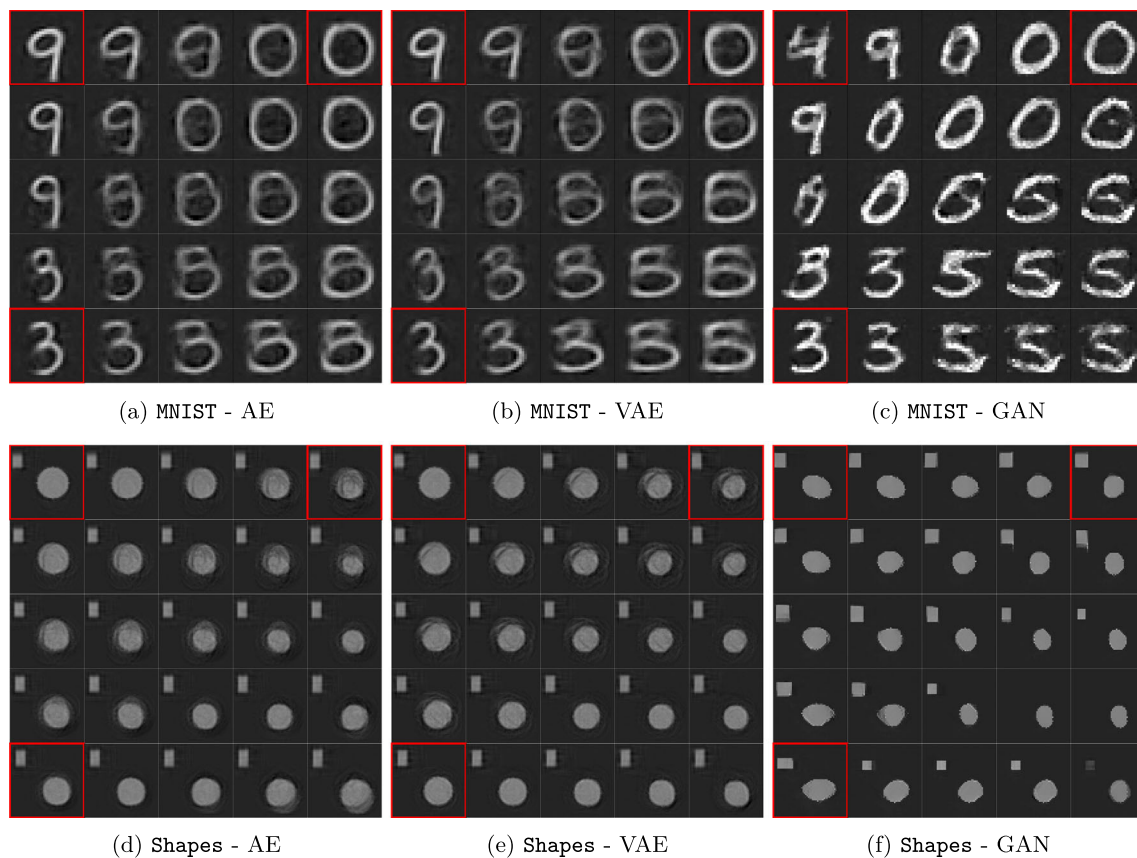
Fig. 6 Interpolation ability of an AE, VAE and GAN. The highlighted top left, bottom left and top right latent space values were chosen close to the test dataset and the other images are computed via linear combinations in the latent space (Color figure online)

5.1 Deconvolution

Figure 7 shows solutions to the deconvolution inverse problem with added Gaussian noise (standard deviation $\sigma = 0.1$) on the MNIST dataset. We test the relaxed, hard and sparse methods with a GAN against Tikhonov and try a range of regularisation parameters. Each reconstruction used the same realisation of noise affecting the data. Hard gives good PSNR results despite not reaching the Morozov discrepancy value, the expected value of the L2 norm of the added Gaussian noise [88]. For the hard constraints reconstructions are restricted to the range of the generator and we do not expect the data discrepancy to go to zero as λ decreases. In the relaxed and sparse constrained reconstructions, for smaller values of λ the solutions tend towards a least squares solution which fits the noise and is affected by the ill-posedness of the inverse problem. The additional variation in the choice of μ, as shown by the additional coloured dots, has little effect on smaller values of λ.

Figure 8 again shows a deconvolution problem with added Gaussian noise (standard deviation $\sigma = 0.1$) on the MNIST dataset. We choose the hard reconstruction for the three different generator models and show three random initial-

isations in $\mathcal{Z}$. Regularisation parameters were chosen to maximise PSNR. The best results are given by the AE and the VAE. The GAN has failed to find a good value in the latent space to reconstruct the number three. The choice of the initial value of z significantly affects the outcome of the reconstruction in the GAN case.

5.2 Compressed Sensing

Consider the compressed sensing inverse problem ($m = 150$ measurements) with added Gaussian noise (standard deviation $\sigma = 0.05$) on MNIST images. We choose regularisation parameters that optimise PSNR over 20 test images. Figure 9 includes a table with the PSNR results on an additional 100 test images. Due to the cartoon-like nature of the MNIST digits, TV regularisation is particularly suitable, however, VAE and AE hard and VAE relaxed are competitive with TV. For more context, example plots for the VAE and TV reconstructions are given in Fig. 9.

To give an indication of computational cost, Tikhonov reconstruction on the compressed sensing inverse problem on the MNIST dataset took on average 32 iterations of backtracking until the relative difference between iterates was
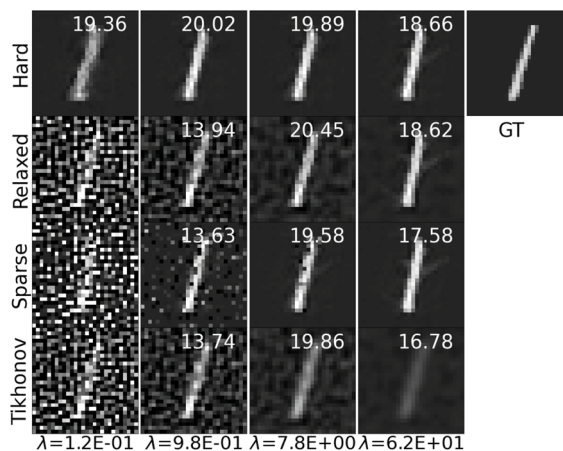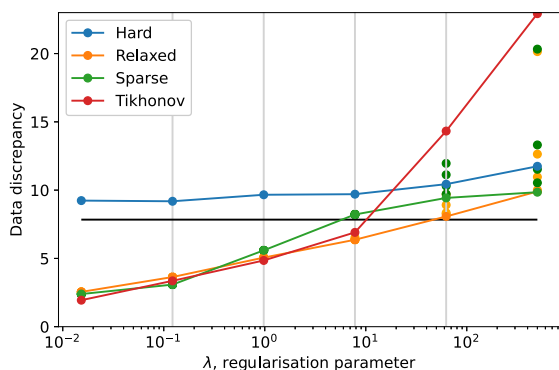
Fig. 7 Solution of the deconvolution problem on MNIST with an eight-dimensional GAN. The plot shows the L2 reconstruction loss against regularisation parameter choice λ in comparison with the Morozov dis-crepancy value in black. Differing choices for μ are plotted as additional markers. The image plots correspond to the parameter values shown by the grey lines and include the PSNR values (Color figure online)
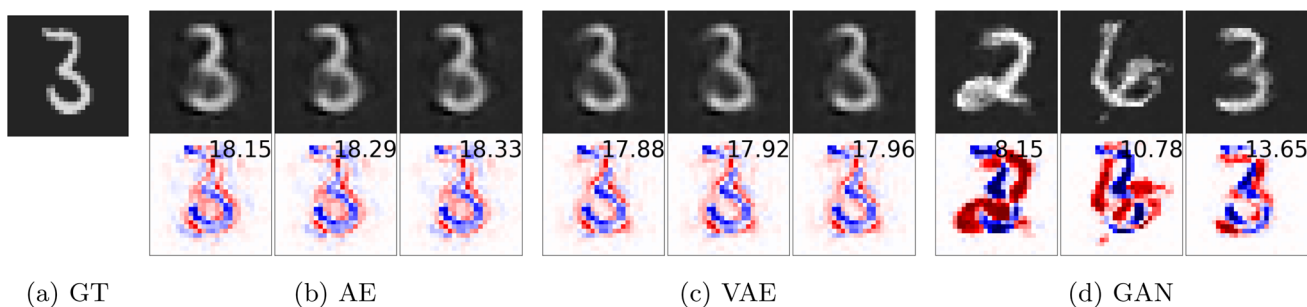

(a) GT   (b) AE   (c) VAE   (d) GAN

Fig. 8 Comparisons between the three generators, with eight-dimensional latent space, for the deconvolution problem. Reconstructions use the hard method. The plot shows 3 different initialisations for each generator. The ground truth (GT) is given on the left, the top line shows the reconstruction and the bottom line the residuals with the PSNR values (Color figure online)

less than 10^{-8}. In comparison, the hard and relaxed took on average 54 and 325 backtracking steps, respectively, without random restarts. TV reconstruction uses a primal-dual hybrid gradient optimisation scheme and is not directly comparable. The algorithms took up to 1 s for Tikhonov, 5 s for hard and 10 s for relaxed.
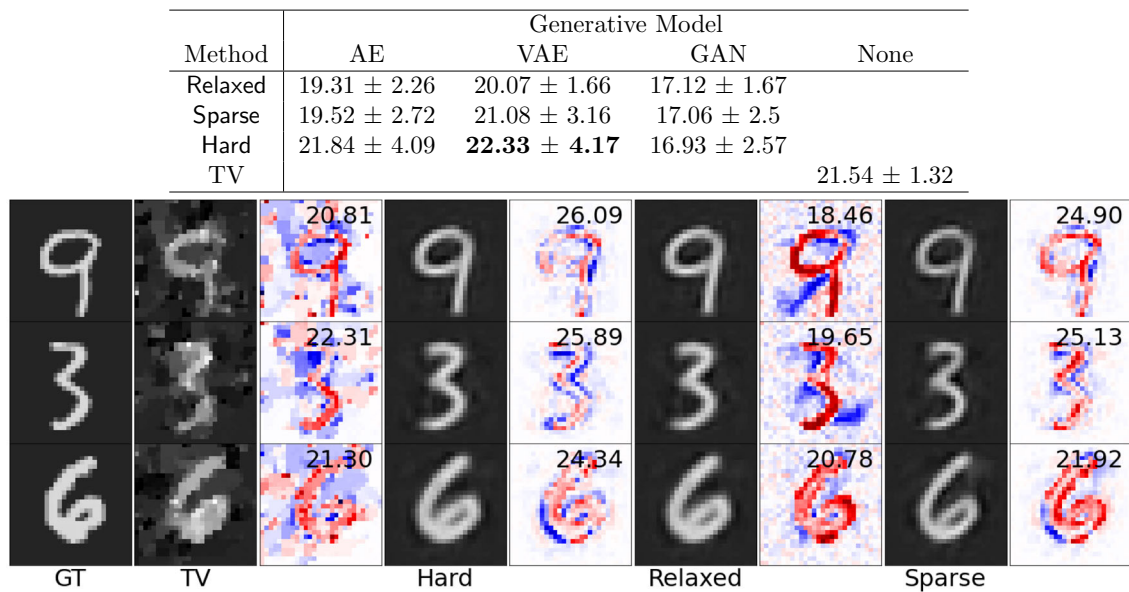
5.3 Tomography

Taking the tomography inverse problem with added Gaussian noise (standard deviation $\sigma = 0.1$), Fig. 10 includes a table which gives the average and standard deviation for the PSNR of 100 reconstructed Shapes images. The regularisation parameters were set to maximise the PSNR over a separate dataset of 20 test images. The GAN has a particularly poor performance but the AE and VAE results are all competitive with TV. Example reconstructions for the AE methods and TV reconstruction are given in Fig. 10. The generative regulariser gives a clear rectangle and circle while the TV reconstruction gives shapes with unclear

outlines and blob-like artefacts. In terms of computational cost, Tikhonov took on average 157 iterations of backtracking until the relative difference between iterates was less than 10^{-8}. In comparison, the hard and relaxed took on average 37 and 255 iterations, respectively, without random restarts. The algorithms took up to 40 s for Tikhonov, up to 12 s for hard and up to 60 s for relaxed.
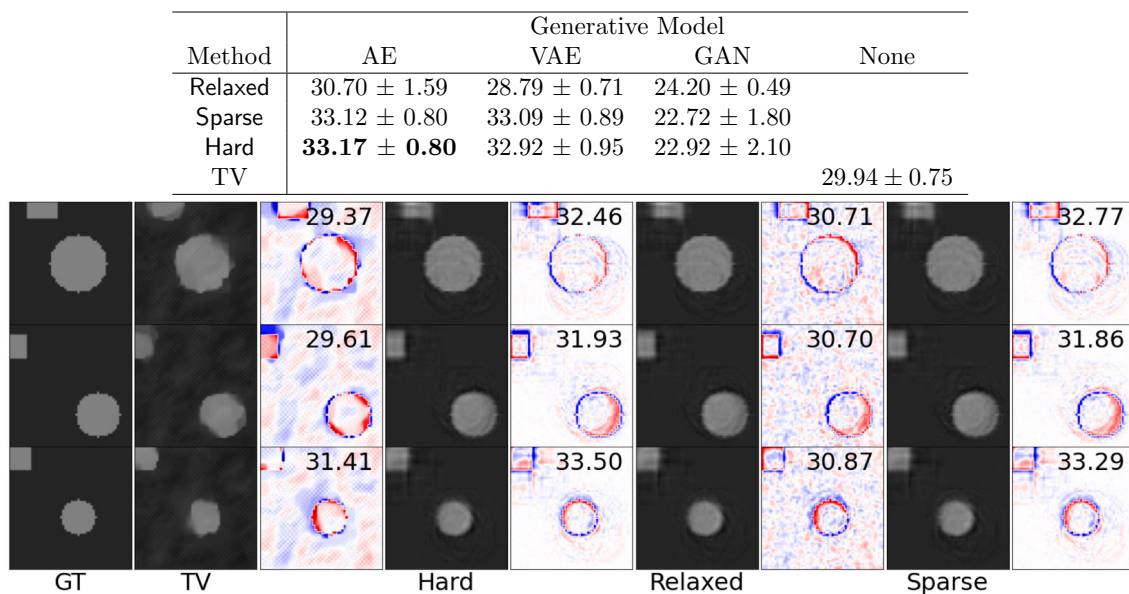
5.4 Out-of-Distribution Testing

We augment the Shapes dataset, creating a shapes+ dataset, with the addition of a bright spot randomly located in the circle. We then take the Tomography inverse problem on the shapes+ dataset with added Gaussian noise (standard deviation $\sigma = 0.05$). For a generative regulariser we use sparse, with $F(x) = \|\nabla x\|_1$, the TV norm. Crucially, the VAE generator used was trained only on the standard Shapes dataset, without bright spots. We compare to standard TV reconstruction. The regularisation parameters were chosen to maximise the PSNR on 20 ground truth and recon-

	Generative Model			
Method	AE	VAE	GAN	None
Relaxed	19.31 ± 2.26	20.07 ± 1.66	17.12 ± 1.67	
Sparse	19.52 ± 2.72	21.08 ± 3.16	17.06 ± 2.5	
Hard	21.84 ± 4.09	$\mathbf{22.33 \pm 4.17}$	16.93 ± 2.57	
TV				21.54 ± 1.32

Fig. 9 Results compare the three different regularisers and three different methods against the unlearned TV reconstruction on the compressed sensing inverse problem. The table shows the mean and standard deviations of PSNR values of 100 reconstructions. The plots show three example solutions, comparing the VAE reconstructions to TV reconstruction. The left column shows the ground truth, even columns the reconstructed images and odd columns the residuals with the PSNR values (Color figure online)

	Generative Model			
Method	AE	VAE	GAN	None
Relaxed	30.70 ± 1.59	28.79 ± 0.71	24.20 ± 0.49	
Sparse	33.12 ± 0.80	33.09 ± 0.89	22.72 ± 1.80	
Hard	$\mathbf{33.17 \pm 0.80}$	32.92 ± 0.95	22.92 ± 2.10	
TV				29.94 ± 0.75

Fig. 10 Results compare the three different regularisers and three different methods against the unlearned TV reconstruction on the tomography inverse problem. The table shows the mean and standard deviations of PSNR values of 100 reconstructions. The plots show five example reconstructions, comparing the AE and TV reconstructions. The left column shows the ground truth, even columns the reconstructions and odd columns the residuals with the PSNR values (Color figure online)

Fig. 11 Tomography inverse problem on random images from the `shapes+` dataset. It compares the use of sparse method, where sparsity is measured in the TV-norm, with a standard TV reconstruction. The generator is a ten-dimensional VAE trained on `Shapes` images. In the final column, the part of the reconstruction lying in the range of the generator is coloured red and the sparse addition is green (Color figure online)
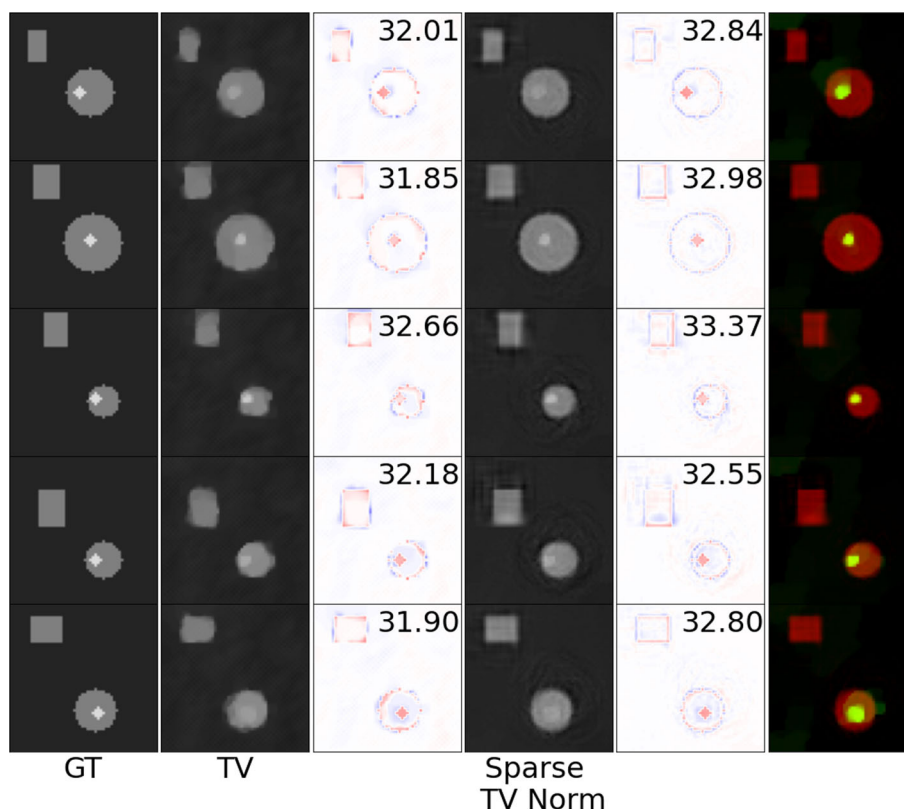
structed images. The mean PSNR over 100 test images for the sparse case is 32.83 with a standard deviation of 0.65 and for the TV reconstruction is 32.01 with a standard deviation of 0.67. Figure 11 shows five reconstructions. The sparse deviations allow reconstruction of the bright spot demonstrating that generative regularisers can also be effective on images close to, but not in, the training distribution.

5.5 `FastMRI` Dataset

We also train a VAE to produce knee `FastMRI` images. The VAE architecture is based on Narnhofer et al. [50]. The `FastMRI` knee dataset contains data 796 fully sampled knee MRI magnitude images [89, 90], without fat suppression. We extract 3,872 training and 800 test ground truth images from the dataset, selecting images from near the centre of the knee, resizing the images to 128×128 pixels and rescaling to the pixel range [0, 1]. The `FastMRI` VAE models took approximately 12 h to train on the same system as above.

Results for the tomography inverse problem with added Gaussian noise of varying standard deviation are given in Fig. 12. For each image and noise level, the same noise instance is used for each reconstruction method, and additionally, for each method, a range of regularisation parameters are tested, and the reconstruction with the best PSNR value is chosen. The plot shows how the PSNR values, averaged over 50 test

images, vary with the noise level. The relaxed method gives the best PSNR values, outperforming Tikhonov although with a larger variance. The sparse method curve has a similar shape to Tikhonov, but performs consistently worse suggesting that this choice of deviations from the generator is not suited to this dataset, generator or inverse problem. We see that for the hard method the results are consistent across the range of noise levels, not improving with reduced noise. The example images reflect the data with the hard reconstruction doing comparatively better with larger noise levels, but the relaxed method captures more of the fine details at the lower noise level.

6 Conclusions and Open Problems

We considered the use of a generator, from a generative machine learning model, as part of the regulariser of an inverse problem, *generative regularisers*. Generative regularisers link the theoretically well-understood methods of variational regularisation with state-of-the-art machine learning models. The trained generator outputs data similar to training data and the regulariser restricts solutions (close) to the range of the generator. The cost of these generative regularisers is in the need for generative model training, the requirement for a large amount of training data and the
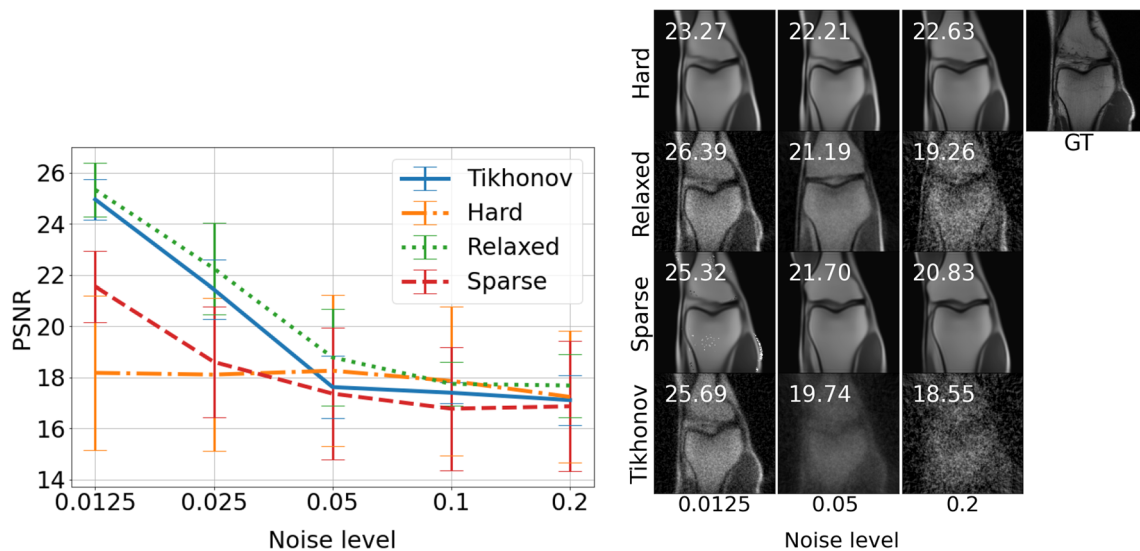
Fig. 12 Reconstructions of the tomography inverse problem with additive Gaussian noise of varying standard deviations. Regularisation parameters are chosen to maximise the PSNR value for each image. The left shows the average PSNR values, taken over 50 test images, with the standard deviation in the error bars and the right shows one particular example reconstruction with PSNR values (Color figure online)

difficulty of the resulting non-convex optimisation scheme. Weighing up the costs and benefits will depend on the inverse problem and the availability of data.

We compared three different types of generative regularisers which either restrict solutions to exactly the range of the generator or allow small or sparse deviations. We found that in simpler datasets the restriction to the range of the generator was successful. Where the ground truth was more complex then allowing small deviations produced the best results. A key benefit of generative regularisers over other deep learning approaches is that paired training data is not required, making the method flexible to changes in the forward problem. We demonstrated the use of generative regularisers on deconvolution and fully sampled tomography problems, both with gradually decaying singular values of the forward operator; and compressed sensing, with a large kernel and non-unique solutions.

The training of the generator is crucial to the success of generative regularisers, and a key contribution of this report is a set of desirable properties for a generator. Numerical tests linked to these properties were discussed and applied to three generative models: AEs, VAEs and GANs. None of these models fulfils the criteria completely. We observed known issues such as mode collapse and discriminator failure in the GAN, blurry images in the VAE and the lack of a prior in the AE. In the inverse problem experiments in this paper, the AE and the VAE yielded the most consistent results. The success of the AE, despite the lack of prior on the latent space, surprised us. We suspect the implicit regularisation of the model from the architecture and initialisations helped make the AE a usable generator. The GAN models did worst in the inverse

problem examples: they generally seemed more sensitive to the initialisation of the non-convex optimisation, making the optimal point in the latent space difficult to recover.

6.1 Future Outlook

We identify three key areas for future growth in the field. Firstly, the desired criteria set out in Sect. 4 for a generative model to be an effective regulariser are useful guidelines but they are not mathematically precise. The field of generative modelling is growing quickly and generators are improving every day. For safety-critical inverse problems, such as medical imaging, being able to assess the quality of a given trained generator quantitatively, would be hugely beneficial. Future work could consider refining these criteria into mathematically precise statements that could be used to assess a given regulariser.

Secondly, we note that the `hard`, `soft` and `sparse` methods are not always regularisers in the strict mathematical definition of a linear regulariser (see definition 4.2 of [3]). For example for $\mathcal{X} = \mathcal{Y} = \mathbb{R}^2$, $A = I$, $\mathcal{Z} = \mathbb{R}$ and $G(z) = (z, z^2)^T$, and observed data $y = (0, 4)^T$, due to the non-convexity of the range of the generative model, for small values of μ, all three regularisers would not give a unique solution to the optimisation task and would be sensitive to noise in the first coordinate of $\mathcal{Y}$. Future work could consider conditions on the generator to indeed yield a convergent regularisation method.

Finally, generative models are currently trained first and subsequently applied to an inverse problem. The benefit of this split approach is that the model does not need retraining

if there are changes in the forward problem. However, future work could consider how to train generative models with inverse problems in mind. For example, linking to the desired criteria, if property B1 was refined to require Lipschitz continuity with a small coefficient, then with suitable choices of architecture and loss function, this could be ensured. Similarly, if convexity of the generator was required, then convex architectures could be considered. With a specific inverse problem in mind, one could also consider training the generator with a penalty for producing images with artefacts or by choosing a loss function to encourage a smooth transition between images from the same subject.

Declarations

Appendix A: Optimisation Algorithms

Algorithm 1 Gradient Descent with Backtracking to solve $\min_z f(z)$.

1: Initialise z_0, $L > 0$, $0 < \eta_0 < 1$, $\eta_1 > 1$.
2: **for** $i = 1, ..., K$ **do**
3:     Let $\tilde{z}(L) := z_{i-1} - \frac{1}{L}\nabla f(z_{i-1})$
4:     **while** $f(\tilde{z}(L)) \geq f(z_{i-1}) - \frac{1}{2L}\|\nabla f(z_{i-1})\|_2^2$ **do**
5:         $L = L\eta_1$
6:     **end while**
7:     $z_i = \tilde{z}$ and $L = L\eta_0$.
8: **end for**

Algorithm 2 Alternating gradient descent with backtracking to solve $\min_{z,x} f(z, x)$.

1: Initialise z_0 and x_0, $L_z > 0, L_x > 0$, $0 < \eta_0 < 1$ and $\eta_1 > 1$
2: **for** $i = 1, ..., K$ **do**
3:     Let $\tilde{z}(L_z) := z_i - \frac{1}{L_z}\nabla f(z_i, x_i)$
4:     **while** $f(\tilde{z}(L_z), x_i) \geq f(z_i, x_i) - \frac{1}{2L_z}\|\nabla f(z_i, x_i)\|_2^2$ **do**
5:         $L_z = L_z\eta_1$
6:     **end while**
7:     Let $z_{i+1} = \tilde{z}(L_z)$ and then $L_z = L_z\eta_0$
8:     Let $\tilde{x}(L_x) := x_i - \frac{1}{L_x}\nabla f(z_{i+1}, x_i)$
9:     **while** $f(z_{i+1}, \tilde{x}(L_x)) \geq f(z_{i+1}, x_i) - \frac{1}{2L_x}\|\nabla f(z_{i+1}, x_i)\|_2^2$ **do**
10:        $L_x = L_x\eta_1$
11:     **end while**
12:     Let $x_{i+1} = \tilde{x}(x_L)$ and $L_x = L_x\eta_0$
13: **end for**

Algorithm 3 PALM with backtracking to solve $\min_{z,u} f(z, u) + g_1(z) + g_2(u)$. Define $\text{prox}_h(z) = \arg\min_x \{h(x) + \frac{1}{2}\|x - z\|_2^2\}$.

1: Initialise z_0, u_0, $L_z > 0$, $L_x > 0$, $0 < \eta_0 < 1$ and $\eta_1 > 1$.
2: **for** $i = 1, ...K$ **do**
3:     Let $\tilde{z}(L_z) := \text{prox}_{\frac{1}{L_z}g_1}(z_i - \frac{1}{L_z}\nabla_z f(z_i, u_i))$
4:     **while** $f(\tilde{z}(L_z), u_i) > f(z_i, u_i) + \nabla_z f(z_i, u_i)^T(\tilde{z}(L_z) - u_i) + \frac{L_z}{2}\|\tilde{z}(L_z) - z_i\|_2^2$ **do**
5:         $L_z = L_z\eta_1$
6:     **end while**
7:     Let $z_{i+1} = \tilde{z}(L_z)$ and then $L_z = L_z\eta_0$
8:     Let $\tilde{u}(L_u) := \text{prox}_{\frac{1}{L_u}g_2}(u_i - \frac{1}{L_u}\nabla_u f(z_{i+1}, u_i))$
9:     **while** $f(z_{i+1}, \tilde{u}(L_u)) > f(z_{i+1}, u_i) + \nabla_u f(z_{i+1}, u_i)^T(\tilde{u}(L_u) - u_i) + \frac{L_u}{2}\|\tilde{u}(L_u) - u_i\|_2^2$ **do**
10:        **end while**
11:     Let $u_{i+1} = \tilde{u}(L_u)$ and then set $L_u = L_u\eta_0$
12: **end for**

Appendix B: Generative Model Architectures

The architectures for the three different generative models, for the different datasets, are given in this Appendix (Figs. 13, 14, 15, 16).

Down-conv: $[f_1, f_2, \dots, f_l]$, $[s_1, s_2, \dots, s_l]$, $[k_1, k_2, \dots, k_l]$, activation, $[d_1, d_2, d_3]$	Convolution with f_1 filters, $k_1 \times k_1$ kernel, stride s_1, activation Dropout layer (prob=0.8) ⋮ Convolution with f_l filters, $k_l \times k_l$ kernel, stride s_l, activation Dropout layer (prob=0.8) Output size=$[d_1, d_2, d_3]$
Up-conv: $[f_1, f_2, \dots, f_l]$, $[s_1, s_2, \dots, s_l]$, $[k_1, k_2, \dots, k_l]$, activation, $[d_1, d_2, d_3]$	Convolution transpose with f_1 filters, $k_1 \times k_1$ kernel, stride s_1, activation Dropout layer (prob=0.8) ⋮ Convolution transpose with f_l filters, $k_l \times k_l$ kernel, stride s_l, activation Dropout layer (prob=0.8) Output size= $[d_1, d_2, d_3]$

Fig. 13 Definitions used in Figs. 14, 15 and 16 (Color figure online)

Encoder/Discriminator			Decoder/Generator		
AE	VAE	GAN	AE	VAE	GAN
Input shape = $[n, n, 1]$			Input shape = [latent dimension]		
			Dense layer, LeakyReLU, $[25]$ Dense layer, LeakyReLU, $[\frac{n^2}{16}]$ Reshape, $[\frac{n}{4}, \frac{n}{4}, 1]$		
Down-conv: $[64, 64, 64]$, $[2, 2, 1]$, $[4, 4, 4]$, LeakyReLU, $[\frac{n}{4}, \frac{n}{4}, 64]$			***Up-conv:*** $[64, 64, 64]$, $[1, 2, 2]$, $[4, 4, 4]$, ReLU, $[n, n, 64]$		
Reshape, $[16n^2]$			Reshape, $[64n^2]$		
Dense layer, [latent dimension]	Dense layer, [2*latent dimension]	Dense layer, [1]	Dense layer, $[n^2]$ Reshape, $[n, n]$		

Fig. 14 The architectures for the 3 generative models, AE, VAE and GAN, for the `MNIST` dataset. The convolution block definitions are given in Fig. 13 (Color figure online)

Fig. 15 — Encoder/Discriminator and Decoder/Generator architectures

Encoder/Discriminator			Decoder/Generator		
AE	VAE	GAN	AE	VAE	GAN
Input shape = $[n, n, 1]$			Input shape = [latent dimension]		
			Dense layer, LeakyReLU, $[\frac{n^2}{64}]$ — Reshape, $[\frac{n}{8}, \frac{n}{8}, 1]$		
Down-conv: $[64, 64, 64]$, $[1, 2, 2]$, $[4, 4, 4]$, LeakyReLU, $[\frac{n}{4}, \frac{n}{4}, 64]$			Up-conv: $[64, 64, 64]$, $[2, 2, 1]$, $[4, 4, 4]$, LeakyReLU, $[n, n, 64]$		
Reshape, $[16n^2]$			Reshape, $[64n^2]$		
Dense layer, [latent dimension]	Dense layer, [2*latent dimension]	Dense layer, [1]	Dense layer, $[n^2]$ — Reshape, $[n, n]$		

Fig. 15 The architectures for the 3 generative models, AE, VAE and GAN, for the Shapes dataset. The convolution block definitions are given in Fig. 13 (Color figure online)

Fig. 16 The architectures for the knee dataset VAE. The convolution block definitions are given in Fig. 13 (Color figure online)

Encoder/Discriminator	Decoder/Generator
VAE	VAE
Input shape = $[n, n, 1]$	Input shape = [latent dimension]
	Dense layer, ReLU, $[\frac{n^2}{8}]$ — Reshape, $[\frac{n}{16}, \frac{n}{16}, 16]$
Down-conv: $[8, 16, 32]$, $[1, 1, 1]$, $[3, 3, 3]$, LeakyReLU, $[n, n, 32]$ Down-conv: $[64, 64, 64]$, $[2, 1, 1]$, $[3, 3, 3]$, LeakyReLU, $[\frac{n}{2}, \frac{n}{2}, 64]$ Down-conv: $[128, 128, 128]$, $[2, 1, 1]$, $[3, 3, 3]$, LeakyReLU, $[\frac{n}{4}, \frac{n}{4}, 128]$ Down-conv: $[256, 256, 256]$, $[2, 1, 1]$, $[3, 3, 3]$, LeakyReLU, $[\frac{n}{8}, \frac{n}{8}, 256]$ Down-conv: $[64, 32, 8]$, $[1, 1, 1]$, $[3, 3, 3]$, LeakyReLU, $[\frac{n}{8}, \frac{n}{8}, 8]$	Up-conv: $[32, 64, 128, 256]$, $[1, 1, 1, 1]$, $[3, 3, 3]$, ReLU, $[\frac{n}{16}, \frac{n}{16}, 256]$ Up-conv: $[512, 512, 512, 512]$, $[1, 1, 1, 1]$, $[3, 3, 3]$, ReLU, $[\frac{n}{16}, \frac{n}{16}, 512]$ Up-conv: $[256, 256, 256, 256]$, $[2, 1, 1, 1]$, $[3, 3, 3]$, ReLU, $[\frac{n}{8}, \frac{n}{8}, 256]$ Up-conv: $[128, 128, 128, 128]$, $[2, 1, 1, 1]$, $[3, 3, 3]$, ReLU, $[\frac{n}{4}, \frac{n}{4}, 128]$ Up-conv: $[64, 64, 64, 64]$, $[2, 1, 1, 1]$, $[3, 3, 3, 3]$, ReLU, $[\frac{n}{2}, \frac{n}{2}, 64]$ Up-conv: $[32, 16, 8, 4]$, $[2, 1, 1, 1]$, $[3, 3, 3, 3]$, ReLU, $[n, n, 8]$
Dense layer, [2*latent dimension]	Up-conv: $[1]$, $[1]$, $[3]$, ReLU, $[n, n, 1]$

References

1. Scherzer, O., Grasmair, M., Grossauer, H., Haltmeier, M., Lenzen, F.: Variational regularization methods for the solution of inverse problems. In: Variational Methods in Imaging (2019)
2. Ito, K., Jin, B.: Inverse Problems: Tikhonov Theory And Algorithms. Applied Mathematics, World Scientific, Singapore (2014)
3. Benning, M., Burger, M.: Modern regularization methods for inverse problems. Acta Numer. **27**, 1–111 (2018)
4. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Phys. D **60**, 259–268 (1992)
5. Aharon, M., Elad, M., Bruckstein, A.: K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. Signal Process. **54**, 4311–4322 (2006)
6. Venkatakrishnan, S.V., Bouman, C.A., Wohlberg, B.: Plug-and-play priors for model based reconstruction. In: GlobalSIP, pp. 945–948 (2013)
7. Meinhardt, T., Moeller, M., Hazirbas, C., Cremers, D.: Learning proximal operators: using denoising networks for regularizing inverse imaging problems. In: ICCV, pp. 1799–1808 (2017)
8. Romano, Y., Elad, M., Milanfar, P.: The little engine that could: regularization by denoising (red). SIAM J. Imaging Sci. **10**, 1804–1844 (2017)
9. Li, H., Schwab, J., Antholzer, S., Haltmeier, M.: NETT: solving inverse problems with deep neural networks. Inverse Probl. **36**, 065005 (2020)
10. Obmann, D., Nguyen, L., Schwab, J., Haltmeier, M.: Sparse Anett for solving inverse problems with deep learning. In: International Symposium on Biomedical Imaging Workshops, Proceedings (2020)
11. González, M., Almansa, A., Delbracio, M., Musé, P., Tan, P.: Solving inverse problems by joint posterior maximization with a VAE prior. SIAM J. Imaging Sci. **15**, 822–859 (2022)
12. Lunz, S., Öktem, O., Schönlieb, C.B.: Adversarial regularizers in inverse problems. In: NeurIPS, pp. 8507–8516 (2018)

13. Arridge, S., Maass, P., Öktem, O., Schönlieb, C.B.: Solving inverse problems using data-driven models. Acta Numer **28**, 1–174 (2019)
14. Bora, A., Jalal, A., Price, E., Dimakis, A.G.: Compressed sensing using generative models. In: ICML, pp. 822–841 (2017)
15. Dhar, M., Grover, A., Ermon, S.: Modeling sparse deviations for compressed sensing using generative models. In: ICML, vol. 3, pp. 1990–2005 (2018)
16. Habring, A., Holler, M.: A generative variational model for inverse problems in imaging. SIAM J. Math. Data Sci. **4**, 306–335 (2022)
17. Tripathi, S., Lipton, Z.C., Nguyen, T.Q.: Correction by projection: denoising images with generative adversarial networks. ArXiv Preprint arXiv:1803.04477 (2018)
18. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016). http://www.deeplearningbook.org
19. Boink, Y.E., Brune, C.: Learned SVD: solving inverse problems via hybrid autoencoding. ArXiv Preprint arXiv:1912.10840 (2019)
20. Obmann, D., Schwab, J., Haltmeier, M.: Deep synthesis regularization of inverse problems. ArXiv Preprint arXiv:2002.00155 (2020)
21. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: ICML, pp. 298–321 (2017)
22. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of wasserstein GANs. NeurIPS, pp. 5768–5778 (2017)
23. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NeurIPS, pp. 2672–2680 (2014)
24. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: ICLR (2014)
25. Kingma, D.P., Welling, M.: An introduction to variational autoencoders. Found. Trends Mach. Learn. **12**, 307–392 (2019)
26. Dave, A., Vadathya, A.K., Subramanyam, R., Baburajan, R., Mitra, K.: Solving inverse computational imaging problems using deep pixel-level prior. IEEE Trans. Comput. Imaging **5**, 37–51 (2018)
27. Jacobsen, J.H., Smeulders, A., Oyallon, E.: i-RevNet: deep invertible networks. In: ICLR (2018)
28. Kingma, D.P., Dhariwal, P.: Glow: generative flow with invertible 1x1 convolutions. In: NeurIPS, pp. 10215–10224 (2018)
29. Oberlin, T., Verm, M.: Regularization via deep generative models: an analysis point of view. In: ICIP, pp. 404–408 (2021)
30. Song, Y., Ermon, S.: Improved techniques for training score-based generative models. In: NeurIPS (2020)
31. Ramzi, Z., Remy, B., Lanusse, F., Starck, J.-L., Ciuciu, P.: Denoising score-matching for uncertainty quantification in inverse problems. In: NeurIPS (2020)
32. Jalal, A., Arvinte, M., Daras, G., Price, E., Dimakis, A.G., Tamir, J.I.: Robust compressed sensing mri with deep generative priors. In: NeurIPS (2021)
33. Behrmann, J., Vicol, P., Wang, K.-C., Grosse, R., Jacobsen, J.-H.: Understanding and mitigating exploding inverses in invertible neural networks. In: AISTATS, PMLR, vol. 130, pp. 1792–1800 (2021)
34. Gu, J., Shen, Y., Zhou, B.: Image processing using multi-code GAN prior. In: CVPR, pp. 3009–3018 (2020)
35. Mosser, L., Dubrule, O., Blunt, M.J.: Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. Math. Geosci. **52**, 53–79 (2020)
36. Chandramouli, P., Gandikota, K.V., Goerlitz, A., Kolb, A., Moeller, M.: Generative models for generic light field reconstruction. In: TPAMI (2020)
37. Asim, M., Shamshad, F., Ahmed, A.: Blind image deconvolution using deep generative priors. IEEE Trans. Comput. Imaging **6**, 1493–1506 (2020)
38. Hand, P., Leong, O., Voroninski, V.: Phase retrieval under a generative prior. In: NeurIPS, pp. 9136–9146 (2018)
39. Hand, P., Voroninski, V.: Global guarantees for enforcing deep generative priors by empirical risk. IEEE Trans. Inf. Theory **66**, 401–418 (2020)
40. Lei, Q., Jalal, A., Dhillon, I.S., Dimakis, A.G.: Inverting deep generative models, one layer at a time. In: NeurIPS, vol. 32 (2019)
41. Daskalakis, C., Rohatgi, D., Zampetakis, M.: Constant-expansion suffices for compressed sensing with generative priors. In: NeurIPS (2020)
42. Shah, V., Hegde, C.: Solving linear inverse problems using GAN priors: an algorithm with provable guarantees. In: ICASSP, pp. 4609–4613 (2018)
43. Jagatap, G., Hegde, C.: Algorithmic guarantees for inverse imaging with untrained network priors. In: NeurIPS, vol. 32 (2019)
44. Candes, E.J., Tao, T.: Decoding by linear programming. IEEE Trans. Inf. Theory **51**, 4203–4215 (2005)
45. Peng, P., Jalali, S., Yuan, X.: Auto-encoders for compressed sensing. In: NeurIPS (2019)
46. Hegde, C.: Algorithmic aspects of inverse problems using generative models. In: 56th Annual Allerton Conference on Communication, Control, and Computing, pp. 166–172 (2019)
47. Menon, S., Damian, A., Hu, S., Ravi, N., Rudin, C.: Pulse: self-supervised photo upsampling via latent space exploration of generative models. In: CVPR, pp. 2437–2445 (2020)
48. Daras, G., Dean, J., Jalal, A., Dimakis, A.G.: Intermediate layer optimization for inverse problems using deep generative models. In: ICML (2021)
49. Gunn, S., Cocola, J., Hand, P.: Regularized training of intermediate layers for generative models for inverse problems. arXiv preprint arXiv:2203.04382 (2022)
50. Narnhofer, D., Hammernik, K., Knoll, F., Pock, T.: Inverse GANs for accelerated MRI reconstruction. In: SPIE—The International Society for Optical Engineering, p. 45 (2019)
51. Hussein, S.A., Tirer, T., Giryes, R.: Image-adaptive GAN based reconstruction. In: AAAI, pp. 3121–3129 (2019)
52. White, T.: Sampling generative networks. arXiv:1609.04468 (2016)
53. Dai, B., Wipf, D.: Diagnosing and enhancing VAE models. In: ICLR (2019)
54. Bauer, M., Mnih, A.: Resampled priors for variational autoencoders. In: PMLR, pp. 66–75 (2020)
55. Veen, D.V., Jalal, A., Soltanolkotabi, M., Price, E., Vishwanath, S., Dimakis, A.G.: Compressed sensing with deep image prior and learned regularization. ArXiv Preprint (2018)
56. Yeh, R.A., Chen, C., Lim, T.Y., Schwing, A.G., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with deep generative models. In: CVPR, pp. 6882–6890 (2017)
57. Lahiri, A., Jain, A.K., Nadendla, D., Biswas, P.K.: Faster unsupervised semantic inpainting: a GAN based approach. In: ICIP, pp. 2706–2710 (2019)
58. Anirudh, R., Thiagarajan, J.J., Kailkhura, B., Bremer, T.: MimicGAN: robust projection onto image manifolds with corruption mimicking. In: IJCV (2020)
59. Adler, J., Öktem, O.: Deep Bayesian inversion. ArXiv Preprint (2018)
60. Park, H.S., Baek, J., You, S.K., Choi, J.K., Seo, J.K.: Unpaired image denoising using a generative adversarial network in X-ray CT. IEEE Access **7**, 110414–110425 (2019)
61. Yang, G., Yu, S., Dong, H., Slabaugh, G., Dragotti, P.L., Ye, X., Liu, F., Arridge, S., Keegan, J., Guo, Y., Firmin, D.: Dagan: deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction. IEEE Trans. Med. Imaging **37**, 1310–1321 (2018)
62. Lv, J., Zhu, J., Yang, G.: Which GAN? a comparative study of generative adversarial network-based fast MRI reconstruction. Philos. Trans. R. Soc. A Math. Phys. Eng. Sci. **379**, 20200203 (2021)

63. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV, pp. 2242–2251 (2017)

64. Oh, G., Sim, B., Chung, H.J., Sunwoo, L., Ye, J.C.: Unpaired deep learning for accelerated MRI using optimal transport driven Cycle-GAN. IEEE Trans. Comput. Imaging **6**, 1285–1296 (2020)

65. Sim, B., Oh, G., Ye, J.C.: Optimal transport structure of CycleGAN for unsupervised learning for inverse problems. In: ICASSP, pp. 8644–8647 (2020)

66. Kabkab, M., Samangouei, P., Chellappa, R.: Task-aware compressed sensing with generative adversarial networks. In: AAAI, pp. 2297–2304 (2018)

67. Gupta, H., McCann, M.T., Donati, L., Unser, M.: CryoGAN: a new reconstruction paradigm for single-particle cryo-EM via deep adversarial learning. IEEE Trans. Comput. Imaging **7**, 759–774 (2021)

68. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Deep image prior. Int. J. Comput. Vis. **128**, 1867–1888 (2020)

69. Dittmer, S., Kluth, T., Maass, P., Baguer, D.O.: Regularization by architecture: a deep prior approach for inverse problems. J. Math. Imaging Vis. **62**, 456–470 (2020)

70. Borji, A.: Pros and cons of GAN evaluation measures. Comput. Vis. Image Underst. **179**, 41–65 (2019)

71. Theis, L., Oord, A.V.D., Bethge, M.: A note on the evaluation of generative models. In: ICLR (2016)

72. Rubner, Y., Tomasi, C., Guibas, L.J.: Metric for distributions with applications to image databases. In: ICCV, pp. 59–66 (1998)

73. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS, pp. 6627–6638 (2017)

74. Gretton, A., Borgwardt, K.M., Rasch, M.J., Smola, A., Schölkopf, B., Gretton, A.S.: A kernel two-sample test. JMLR **13**, 723–773 (2012)

75. Lopez-Paz, D., Oquab, M.: Revisiting classifier two-sample tests. In: ICLR (2017)

76. Arora, S., Risteski, A., Zhang, Y.: Do GANs learn the distribution? Some theory and empirics. In: ICLR, pp. 1–16 (2018)

77. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2323 (1998)

78. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training GANs. In: NeurIPS, pp. 2234–2242 (2016)

79. GoogleResearch: TensorFlow: large-scale machine learning on heterogeneous systems. Software available from tensorflow.org

80. Bredies, K., Lorenz, D.: Mathematical Image Processing, pp. 1–469. Springer, Berlin (2018)

81. Flamary, R., Courty, N.: POT python optimal transport library. J. Mach Learn. Res. **22**, 3571–3578 (2021)

82. Censor, Y.: The Mathematics of Computerized Tomography, vol. 18, p. 283. SIAM, New Delhi (2002)

83. Davenport, M.A., Duarte, M.F., Eldar, Y.C., Kutyniok, G.: Introduction to Compressed Sensing

84. Adler, J., Kohr, H., Öktem, O.: Operator Discretization Library (ODL)

85. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, É.: Scikit-learn: machine learning in python. JMLR **12**, 2825–2830 (2011)

86. Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. J. Math. Imaging Vis. **40**, 120–145 (2011)

87. Bolte, J., Sabach, S., Teboulle, M.: Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Math. Program. **146**, 459–494 (2014)

88. Tikhonov, A.N., Goncharsky, A.V., Stepanov, V.V., Yagola, A.G.: Numerical Methods for the Solution of Ill-Posed Problems. Kluwer Academic, Netherlands (1995)

89. Knoll, F., Zbontar, J., Sriram, A., Muckley, M.J., Bruno, M., Defazio, A., Parente, M., Geras, K.J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdzalv, M., Romero, A., Rabbat, M., Vincent, P., Pinkerton, J., Wang, D., Yakubova, N., Owens, E., Zitnick, C.L., Recht, M.P., Sodickson, D.K., Lui, Y.W.: fastMRI: a publicly available raw k-space and DICOM dataset of knee images for accelerated MR image reconstruction using machine learning. Radiol. Artif. Intell. **2**, 190007 (2020)

90. Zbontar, J., Knoll, F., Sriram, A., Murrell, T., Huang, Z., Muckley, M.J., Defazio, A., Stern, R., Johnson, P., Bruno, M., Parente, M., Geras, K.J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdzal, M., Romero, A., Rabbat, M., Vincent, P., Yakubova, N., Pinkerton, J., Wang, D., Owens, E., Zitnick, C.L., Recht, M.P., Sodickson, D.K., Lui, Y.W.: fastMRI: an open dataset and benchmarks for accelerated MRI. ArXiv Preprint (2018)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Margaret Duff received an MMath and BA in Mathematics from Emmanuel College, Cambridge. She then went on to complete a PhD at the University of Bath EPSRC Centre for Doctoral Training in Statistical Applied Mathematics (SAMBa) researching generative models applied to inverse imaging problems. She is now working at the Science and Technology Facilities Council, as part of UK Research and Innovation, as a Computational Scientist in Imaging as part of Tomography Group in the Scientific Computing Department. Her research interests include inverse problems in imaging, optimisation and machine learning.

Neill Campbell is a Royal Society Industry Fellow and Professor of Visual Computing and Machine Learning at the University of Bath. He is the Director of the UKRI Centre for the Analysis of Motion, Entertainment Research and Applications (CAMERA) and the Chair of the British Machine Vision Association (the UK body for computer vision). He also holds an Honorary position in the Vision and Imaging Science Group in the Department of Computer Science at University College London where he was formerly a Research Associate. He completed his PhD in Computer Vision in the Machine Intelligence Laboratory in the Department of Engineering at the University of Cambridge. His research involves learning models of shape, appearance and dynamics from images and video. In particular, creating systems that do not require technical computing expertise (e.g. for artists). He also works

on machine learning problems where data are scarce or expensive to obtain (e.g. annotations from expert clinicians) and when uncertainty in the resulting output is important (e.g. medical and safety applications).

Matthias J. Ehrhardt received the Diploma degree (Hons.) in industrial mathematics from the University of Bremen, Germany, in 2011, and the Ph.D. degree in medical imaging from University College London, U.K., in 2015. He held a postdoctoral position with the Cambridge Image Analysis group, Department for Applied Mathematics and Theoretical Physics, University of Cambridge, U.K., from 2016 to 2018. He moved to the University of Bath, U.K. as a Prize Fellow in 2018, where since 2021 he is a Reader at the Department of Mathematical Sciences. He is heading the Bath Imaging Group, a co-director of the Bath Centre for Mathematics and Algorithms for Data and the deputy director of the EPSRC Programme Grant on the Mathematics of Deep Learning. His research interests include optimisation, inverse problems, computational imaging, and machine learning.