Differentiable strong lensing

Uniting gravity and neural nets through differentiable probabilistic programming

Chianese, M.; Coogan, A.; Hofma, P.; Otten, S.; Weniger, C.

DOI
10.1093/mnras/staa1477

Publication date
2020

Document Version
Final published version

Published in
Monthly Notices of the Royal Astronomical Society

Citation for published version (APA):
https://doi.org/10.1093/mnras/staa1477
Differentiable strong lensing: uniting gravity and neural nets through differentiable probabilistic programming

Marco Chianese,1,2 Adam Coogan,1,2 Paul Hofma,1,2 Sydney Otten1,2 and Christoph Weniger1

1Gravitation Astroparticle Physics Amsterdam (GRAPPA), Institute for Theoretical Physics Amsterdam and Delta Institute for Theoretical Physics, University of Amsterdam, Science Park 904, NL-1098 XH Amsterdam, The Netherlands
2Institute for Mathematics, Astrophysics and Particle Physics (IMAPP), Radboud University, Heyendaalseweg 135, NL-6525 AJ Nijmegen, The Netherlands

Accepted 2020 May 19. Received 2020 May 4; in original form 2019 November 15

ABSTRACT

Since upcoming telescopes will observe thousands of strong lensing systems, creating fully automated analysis pipelines for these images becomes increasingly important. In this work, we make a step towards that direction by developing the first end-to-end differentiable strong lensing pipeline. Our approach leverages and combines three important computer science developments: (i) convolutional neural networks (CNNs), (ii) efficient gradient-based sampling techniques, and (iii) deep probabilistic programming languages. The latter automatize parameter inference and enable the combination of generative deep neural networks and physics components in a single model. In the current work, we demonstrate that it is possible to combine a CNN trained on galaxy images as a source model with a fully differentiable and exact implementation of gravitational lensing physics in a single probabilistic model. This does away with hyperparameter tuning for the source model, enables the simultaneous optimization of nearly 100 source and lens parameters with gradient-based methods, and allows the use of efficient gradient-based posterior sampling techniques. These features make this automated inference pipeline potentially suitable for processing a large amount of data. By analysing mock lensing systems with different signal-to-noise ratios, we show that lensing parameters are reconstructed with per cent-level accuracy. More generally, we consider this work as one of the first steps in establishing differentiable probabilistic programming techniques in the particle astrophysics community, which have the potential to significantly accelerate and improve many complex data analysis tasks.

Key words: gravitational lensing: strong – galaxies: structure – dark matter.

1 INTRODUCTION

Strong lensing is a gravitational effect through which an astrophysical light source is observed in distorted, multiple images in the sky due to the deflection of its light by matter distributed along the line of sight (Treu 2010). It has become one of the main ways to probe the small-scale structure of dark matter (DM) haloes, since subhaloes with mass below $\sim 10^8 M_\odot$ do not host stars and are thus invisible (Fitts et al. 2017; Read et al. 2017). The detection (or non-detection) of these subhaloes is a critical tool for discriminating among different paradigms of DM (Bertone, Hooper & Silk 2005; Bertone & Tait 2018). In the standard $\Lambda$-cold dark matter ($\Lambda$CDM) cosmological model, the large-scale structures of the Universe form through the collapse of primordial density fluctuations. The matter content of the Universe is dominated by non-relativistic and almost collisionless substance dubbed CDM. In this scenario, an abundance of small DM substructures is formed, as confirmed by ab-initio $N$-body cosmological simulations (Kuhlen, Vogelsberger & Angulo 2012). On the other hand, alternative well-motivated particle DM scenarios such as warm dark matter (Bode, Ostriker & Turok 2001; Lovell et al. 2014) and self-interacting DM (Vogelsberger et al. 2018; Kahlhoefer et al. 2019) predict a lower abundance of low-mass DM substructures. Recent analyses of strong gravitational lensing of extended sources (Vegetti et al. 2010; Vegetti et al. 2012; Hezaveh et al. 2016; Vegetti et al. 2018; Ritondale et al. 2019) and quasars (Fadely & Keeton 2012; Gilman et al. 2019b, a) have already demonstrated sensitivity to DM haloes with masses larger than $10^9 M_\odot$. In the near future, new observatories like DES (Abbott et al. 2016), LSST (LSST Science Collaboration et al. 2009; Drlica-Wagner et al. 2019; Verma...
et al. 2019), Euclid (Refregier et al. 2010), and next-generation observatories like ELT (Simon et al. 2019) will observe thousands of strong lensing systems with very high precision, pushing the sensitivity of lensing probes of DM substructures to even lower masses. Moreover, these observations will be dominated by lenses at high redshift, increasing the likelihood of detecting small DM haloes along the line of sight (Despali et al. 2018).

In analysing a galaxy–galaxy strong lens, the observed lensed image is reconstructed by simultaneously modelling the surface brightness of the source and the matter distribution of the lens galaxy. This requires parametrizing both of these components. While N-body simulations show that the density profiles of galactic DM haloes are well described by various analytic profiles (Navarro, Frenk & White 1996), the distribution of the source’s light is more complicated. The Sérsic brightness profile (Sérsic 1963) is a common choice (see e.g. Brewer et al. 2011), but is inadequate for high-resolution observations and modelling high-redshift source galaxies, which generally have more complex morphologies than low-redshift ones. Another class of methods computes the source brightness profile on a grid by linearly inverting the observed lensed image given a fixed lens model (Warren & Dye 2003; Suyu et al. 2006). This requires using specific prior with a particular form to regularize the source, depending on two-point quantities calculated between pairs of pixels as well as hyperparameters. These methods can be cast in a fully Bayesian framework and performed on an adaptive grid in the source plane (Koopmans 2005; Veggetti & Koopmans 2009). The public code PYAUTOLENS implements this analysis strategy (Nightingale, Dye & Massey 2018; Nightingale et al. 2019). Extensions of these methods include grid-free approaches using radial basis functions centred on image pixels ray traced back to the source plane (Merten 2016) and methods decomposing the source as a sum of shapelets (Birrer, Amara & Refregier 2015). These methods are available in the SAWLENS2 (Merten 2017) and LENSTRONOMY (Birrer & Amara 2018; Birrer 2019) software packages. Depending on the pipeline’s fitting scheme, the choice of priors for the lensing system’s parameters may be restricted, making it challenging to perform sensitivity analysis. These analysis pipelines also generally require dedicated, time-consuming hyperparameter optimization efforts for fitting each strong lensing system. On the other hand, fully automated lensing pipelines which do not require these interventions will become increasingly useful for analysing large strong lensing data sets, and for subsequently characterizing the underlying DM subhalo mass function. To this aim, it is of paramount importance to develop automated lens modelling techniques. In this paper, we demonstrate that this is possible through the use of new computational tools from the field of deep learning such as automatic differentiation (AD). This enables the use of powerful gradient descent-based methods which are crucial to speed up and automatize the fitting procedure of high-dimensional parameter space. In particular, this paper focuses on how deep generative models can be combined with known physics using AD to create a new pipeline for analysing images of galaxy–galaxy strong lensing systems.

Deep learning has advanced dramatically over the past decade, with accurate image classifiers (Krizhevsky, Sutskever & Hinton 2012) and a host of generative methods such as variational autoencoders (VAEs, Kingma & Welling 2013; Jimenez Rezende, Mohamed & Wierstra 2014), generative adversarial networks (Goodfellow et al. 2014; Brock, Donahue & Simonyan 2018; Karras, Laine & Aila 2018), and flow-based models (Kingma & Dhariwal 2018) capable of producing novel, realistic images counting among its successes. These methods have also been applied to the physical sciences (Carleo et al. 2019), with topics including deblending galaxy images (Reiman & Göhr 2019), generating weak gravitational lens convergence maps (Mustafa et al. 2019), and classifying LHC jet events (Larkoski, Moult & Nachman 2017). However, there has been considerably less scientific investigation into leveraging AD (Baydin, Pearlmutter & Radul 2015), the core technology enabling training of neural networks with millions of parameters using gradient descent. AD libraries (Paszke et al. 2019; Revels, Lubin & Papamarkou 2016; Innes et al. 2019) make it possible to take exact derivatives of arbitrary computable functions by using the chain rule to compose the gradients of individual operations.

The approach of creating automatically differentiable mechanistic models that can be combined with deep neural networks is known as differentiable programming (Innes et al. 2019). Differentiable programming has recently been applied to engineering problems, with demonstrated benefits for challenging optimization problems in various domains. Constructing a differentiable ray tracer (Li et al. 2018a) and image-processing algorithms (Li et al. 2018b; Sitzmann et al. 2018) simplifies the inverse problem of fitting parameters describing the lighting, materials and objects in a scene from a photograph, as well as the forward problem of optimizing image-processing pipelines. Differentiable rigid-body physics engines make it possible to train deep learning controllers for robots in accurate environments (Degrave et al. 2016). These simulators can also be combined with neural networks to model physical systems based on video and predict their future behaviour (de Avila Belbute-Peres et al. 2018).

Recently, deep learning methods have been brought to bear on strong lensing analyses. Hezaveh, Perreault Levasseur & Marshall (2017) and Perreault Levasseur, Hezaveh & Wechsler (2017) applied a convolutional neural network (CNN, LeCun et al. 1989) to infer the parameters of singular isothermal ellipsoid lenses (Keeton & Kochanek 1998; Kormann, Schneider & Bartelmann 1994) with unprecedented speed. This CNN was later coupled to an optimizer controlled by another CNN that performs a linear inversion to recover the source’s pixelated brightness profile without requiring regularization hyperparameters (Morningstar et al. 2019). Both CNNs were trained using supervised learning on mock lensing datasets. Very recently, supervised CNNs have been trained using toy models of lensing systems containing substructure to differentiate between different DM models (Alexander et al. 2019) and infer parameters in the subhalo mass function (Brehmer et al. 2019). Moreover, in Diaz Rivero & Dvorkin (2019) a supervised CNN trained to classify whether or not a lensing system observation contained detectable substructure, thus identifying images worthy of follow-up observations and analyses.

In this work, we construct the first differentiable programming-based strong lensing analysis pipeline, consisting of a deep generative model for the source galaxy brightness profile and a physics model for the lens. This approach removes the need for manual modelling or hyperparameter optimization and speeds up inference using gradient-based techniques. In particular, we use a VAE (Kingma & Welling 2013; Jimenez Rezende et al. 2014) to learn a parametric description of the source galaxy’s light. In contrast with the aforementioned deep learning strategies for lensing, the VAE is trained in an unsupervised manner on unlensed galaxies; a similar VAE was previously constructed in Ravanbakhsh et al. (2016). The lensing physics is implemented by solving the Poisson equation for analytical models of the lens and external shear. Since our analysis pipeline is modularized, it is possible to change parameters’ priors or include additional lensing effects such as multiple sources and weighted lensing. In particular, the neutral density field in the source plane can be modelled using a Poisson process, as described in (Perreault Levasseur 2019). This technique also enables training of more complex models for the source's density field. The approach of creating automatically differentiable mechanistic models that can be combined with deep neural networks is known as differentiable programming (Innes et al. 2019). Differentiable programming has recently been applied to engineering problems, with demonstrated benefits for challenging optimization problems in various domains. Constructing a differentiable ray tracer (Li et al. 2018a) and image-processing algorithms (Li et al. 2018b; Sitzmann et al. 2018) simplifies the inverse problem of fitting parameters describing the lighting, materials and objects in a scene from a photograph, as well as the forward problem of optimizing image-processing pipelines. Differentiable rigid-body physics engines make it possible to train deep learning controllers for robots in accurate environments (Degrave et al. 2016). These simulators can also be combined with neural networks to model physical systems based on video and predict their future behaviour (de Avila Belbute-Peres et al. 2018).

Recently, deep learning methods have been brought to bear on strong lensing analyses. Hezaveh, Perreault Levasseur & Marshall (2017) and Perreault Levasseur, Hezaveh & Wechsler (2017) applied a convolutional neural network (CNN, LeCun et al. 1989) to infer the parameters of singular isothermal ellipsoid lenses (Keeton & Kochanek 1998; Kormann, Schneider & Bartelmann 1994) with unprecedented speed. This CNN was later coupled to an optimizer controlled by another CNN that performs a linear inversion to recover the source’s pixelated brightness profile without requiring regularization hyperparameters (Morningstar et al. 2019). Both CNNs were trained using supervised learning on mock lensing datasets. Very recently, supervised CNNs have been trained using toy models of lensing systems containing substructure to differentiate between different DM models (Alexander et al. 2019) and infer parameters in the subhalo mass function (Brehmer et al. 2019). Moreover, in Diaz Rivero & Dvorkin (2019) a supervised CNN trained to classify whether or not a lensing system observation contained detectable substructure, thus identifying images worthy of follow-up observations and analyses.

In this work, we construct the first differentiable programming-based strong lensing analysis pipeline, consisting of a deep generative model for the source galaxy brightness profile and a physics model for the lens. This approach removes the need for manual modelling or hyperparameter optimization and speeds up inference using gradient-based techniques. In particular, we use a VAE (Kingma & Welling 2013; Jimenez Rezende et al. 2014) to learn a parametric description of the source galaxy’s light. In contrast with the aforementioned deep learning strategies for lensing, the VAE is trained in an unsupervised manner on unlensed galaxies; a similar VAE was previously constructed in Ravanbakhsh et al. (2016). The lensing physics is implemented by solving the Poisson equation for analytical models of the lens and external shear. Since our analysis pipeline is modularized, it is possible to change parameters’ priors or include additional lensing effects such as multiple sources and
lenses, and line-of-sight haloes and subhaloes without having to retrain the source VAE. The source model can also be improved independently from the rest of the code.

Our pipeline is implemented in the PYTORCH machine learning framework (Paszke et al. 2019), which contains an AD engine and graphical processing unit (GPU)-accelerated functions for array computations. As with the differentiable programming examples mentioned above, pervasive AD makes it straightforward to fit the high-dimensional parameter vector of our lensing model using gradient-based methods. We also exploit the PYRO (Bingham et al. 2018) probabilistic programming language to characterize the uncertainties of our parameter fits. PYRO is capable of sampling model parameters, computing likelihoods, and automatically performing inference with tools such as Markov Chain Monte Carlo (MCMC) and variational inference (Blei, Kucukelbir & McAuliffe 2016; Hoffman et al. 2012) on arbitrary probabilistic models, even those with stochastic control flow. While probabilistic programming has existed for a long time (Lunn et al. 2000, 2009), it has only recently been integrated with AD, making parameter inference possible even for models including neural networks (Tran et al. 2016; Carpenter et al. 2017; Burren et al. 2018; Bingham et al. 2018; Ge, Xu & Ghahramani 2018; Charnock et al. 2019). In the context of high-energy physics, recent work has reframed the SHERPA event generator using probabilistic programming, enabling more efficient simulation of rare events (Baydin et al. 2018; Günes Baydin et al. 2019). In our present work, we again leverage the differentiability of our pipeline by employing Hamiltonian Monte Carlo (HMC, Neal 2012; Betancourt 2017), a gradient-based MCMC method, to efficiently sample from the high-dimension posterior for the lens and source parameters. The unique meshing of an exact physics model with a deep generative source model in an AD-compatible, probabilistic programming framework makes the current paper one of the first differentiable probabilistic programs to our knowledge.

We begin this paper by discussing the VAE model for source galaxies in Section 2, and describing the training procedure and validation tests. In Section 3, we review the physics of strong gravitational lensing, and introduce all the ingredients required to generate lensed images. In particular, we define the lens and the source models considered in this paper. Section 4 delineates our lensing inference pipeline. In Section 5, we test the pipeline on two mock galaxy–lensing systems, and discuss parameter fitting and posterior analysis. We draw our conclusions in Section 6.

2 DEEP GENERATIVE MODELS FOR SOURCE GALAXIES

This section describes how we use a VAE to construct a parametric model for source galaxy light. After reviewing VAEs and explaining the architecture we selected, we describe how ours is trained and present tests validating the performance of the parametric model we use it to construct.

2.1 Variational autoencoders

Natural images (such as those of galaxies) lie on or near a low-dimensional submanifold in the space of all possible images (Belkin 2003). This motivates the concept of probabilistic latent variable models (Bishop 1998), where a datum \( x \) (such as a galaxy image) is related to a latent variable \( z \) through a conditional probability density \( p(x|z) \), and the latent variables are assumed to have a prior density \( p(z) \). We will use this to construct a parametric model for galaxy images, where \( z \) maps on to the mean of the decoding distribution and has prior \( p(z) \).

The VAE was constructed to efficiently approximate models of this form (Kingma & Welling 2013; Jimenez Rezende et al. 2014). It approximates the conditional density \( p(x|z) \) with a decoder \( d_\theta(x|z) \) whose parameters are functions represented by a neural network; \( \theta \) represents the network’s parameters. The VAE also includes an encoder, \( e_\phi(z|x) \), that similarly approximates the conditional distribution \( p(z|x) \) using a neural network with parameters \( \phi \). Fig. 1 illustrates this structure. The encoder and decoder are typically both taken to be diagonal Gaussians:

\[
d_\theta(x|z) = \mathcal{N}(x|\mu_\theta(z), \sigma_\theta)
\]

\[
e_\phi(z|x) = \mathcal{N}(z|\mu_\phi(x), \sigma_\phi(x)).
\]

In our notation, \( \mathcal{N}(\mu, \sigma) \) denotes that \( x \) follows a normal distribution with mean \( \mu \) and standard deviation \( \sigma \). The functions \( \mu_\phi, \mu_\theta, \sigma_\phi, \sigma_\theta \) are implemented using neural networks, and the decoder’s standard deviation \( \sigma_\phi \) is a constant hyperparameter. We identify \( \sigma_\theta \) with the approximate standard deviation of the Gaussian noise in our training data set.

Training VAE requires a data set \( \{x_i\} \) as well as selecting the latent space’s dimensionality and prior \( p(z) \), which is typically taken to be \( \mathcal{N}(0, I) ^{1} \). Ideally, the VAE would be trained by maximizing the marginal likelihood of the training data set, which requires computing the integral

\[
p_\theta(x) = \int dz \, d_\theta(x|z) p(z)
\]

for each training point. Note that the marginal likelihood only depends on the decoder. However, this integral is generally intractable. The difficulty is circumvented by defining an alternative objective function called the evidence lower bound (ELBO), whose derivation is reviewed in Appendix B. The reason the encoder network was introduced is because it is required to compute the ELBO, whose value for each training point is given by

\[
\log p_\theta(x) \geq \sum_i \text{ELBO}(x_i; \theta, \phi) \equiv \mathbb{E}_{e_\phi(z|x)}[\log d_\theta(x|z)] - D_{KL}[e_\phi(z|x)||p(z)].
\]

\( ^{1} \)Several works have studied more complex prior distributions, including learnable ones (Chen et al. 2016; Dilokthanakul et al. 2016; Tomczak & Welling 2017; Alemi et al. 2017; Goyal et al. 2017).

\( ^{2} \)The objective equation (4) is equivalent to the \( \beta \)-VAE (Higgins et al. 2017) objective obtained by taking \( \beta = 1 \) and setting the decoder’s standard deviation to 1.

Figure 1. Diagram of a VAE. The diagram shows how an input image \( x \) is passed through the encoder (red trapezoid) to yield an encoding distribution (red blob) with mean and standard deviation \( \mu_e(x) \) and \( \sigma_e(x) \). A point is then sampled from this distribution (blue dot) and passed through the decoder (blue trapezoid) to yield a reconstructed image.

MNRAS 496, 381–393 (2020)
The first term is related to the quality of the reconstruction obtained by passing an image through the encoder followed by the decoder: maximizing this term improves reconstruction quality. The function $D_{KL}(\cdot \mid \cdot)$ in the second term is the Kullback–Leibler (KL) divergence, which measures the difference between its two argument probability distributions. This term’s maximization drives the averaged encoding distribution to look more like the prior $p(z)$. The VAE is trained to maximize the ELBO with stochastic gradient descent by taking random minibatches of training images. The second term can often be computed analytically, which makes computing its gradient straightforward. A Monte Carlo estimate of the first term can be computed. Appendix C explains how the derivative of this estimate can be computed.

Our VAE architecture choice is influenced by three widespread trends in deep CNN design (Radford, Metz & Chintala 2015): replacing pooling functions with strided convolutions, relying on convolutional layers rather than fully connected ones when possible, and interleaving batch normalization layers (Ioffe & Szegedy 2015). In more detail, the encoder uses five blocks made up of strided convolutions, batch normalization layers, and LeakyReLU (leaky rectified linear unit) activation functions (Maas, Hannun & Ng 2013). The output from these blocks is processed by two separate dense layers, which give the $\mu_e(x)$ and $\sigma_e(x)$, the parameters of the encoding distribution. The decoder is similarly built from five blocks consisting of a transposed convolution, batch normalization layers, and ReLU (rectified linear unit) activation function. The output of last block is passed through a tanh activation function to produce $\mu_d(x)$, the pixel values of which are thus restricted to lie within $[-1, 1]$. The standard deviation of the decoding distribution was set to $\sigma_d = 1/50$, which is the approximate standard deviation of the noise in our training data set. We also studied the impact of making $\sigma_d$ a trainable parameter (as recommended in Dai & Wipf 2019), in which case we find it converges to approximately this value. We used 64 latent-space dimensions. Complete details of the architecture and weight initializations can be found in Appendix D, where we also describe other architectures with which we experimented.

After training the VAE as described in the next subsection, we obtain a parametric model for galaxy images where $z$ corresponds to the image $\mu_e(z)$, and has prior $p(z)$. However, it is a well-known and difficult-to-solve problem that the assumed prior $p(z)$ does not actually match the ‘aggregate prior’ obtained by encoding the full training data set (Jimenez Rezende et al. 2014; Alemi et al. 2017)

$$q_0(z) = \frac{1}{N} \sum_{i=1}^{N} c_p(z|x^{(i)}).$$

This mismatch can cause problems when performing maximum a posteriori (MAP) estimation of $z$ for a given galaxy image. The assumed prior can drag $z$ into unrealistic regions of parameter space far from the location preferred by the likelihood, where the corresponding decoded image $\mu_d(z)$ does not look like a galaxy.

A variety of methods have been proposed to address this problem, including constructing simple priors using $q_0(z)$ (Tomczak & Welling 2017; Otten et al. 2019), fitting $q_0(z)$ with a second VAE after training the primary one (Dai & Wipf 2019), or using normalizing flows (Jimenez Rezende & Mohamed 2015; Kingma et al. 2016; Papamakarios, Pavlakou & Murray 2017; van den Berg et al. 2018). Here, we follow the simpler approach of creating a weakly informative prior for $z$ by fitting a multivariate normal distribution to the set of encoded means of the training data $[\mu_e(x^{(i)})]_{i=1}^{N}$, and rescaling its covariance matrix by a factor of 9. This prior roughly confines the latent variable to realistic regions of the latent space while remaining diffuse enough that the likelihood drives MAP estimates of $z$ from observations.

### 2.2 Training

We construct a data set for training the VAE starting from 56 062 images of galaxies in the COSMOS field taken by the Hubble Space Telescope (Scoville et al. 2007a,b; Koekemoer et al. 2007). These were used for the GREAT3 weak gravitational lensing challenge (Mandelbaum et al. 2014; Team 2019). An estimate of the pixel noise is included for each image, which is assumed to be Gaussian and uncorrelated. Parameters for S´ersic profiles fit to each of the galaxies are also provided. Details about the image processing and parameter fits can be found in appendix E of Mandelbaum et al. (2014).

We discard the small number of galaxies for which the S´ersic fits failed or gave unphysical parameters. The dimensions of the images in the data set vary, so we remove any that are smaller than 64 × 64 pixels. Images with unequal width and height are cropped into squares, and all are then downsampled to 64 × 64 pixels. Finally, many of the galaxies are extremely bright and small, and some are nearly indistinguishable from the pixel noise. In our experiments both of these degraded the quality of the VAE’s reconstructions, in the former case by biasing it towards producing only compact, bright galaxies and in the latter case by degrading the fidelity of the reconstructions. We find a useful, very heuristic way of removing these is to make a cut on the image ‘signal-to-noise’ (S/N) quantity

$$\max(I_{1/4})/\sigma_{\text{noise}},$$

where $\sigma_{\text{noise}}$ is the standard deviation of the pixel noise and $I_{1/4}$ is the image downsampled by a factor of 4. Restricting this quantity to lie between 15 and 50 reduces the data set to 17 543 images. We then split these into a training, test and validation data sets consisting of 15 500, 500, and 1543 images, respectively. This is a fairly small training data set by industrial machine learning standards (Deng et al. 2009), but could be greatly augmented by future astronomical observations.

The training and validation sets are pre-processed by dividing each image by its maximum pixel value. The VAE’s parameters are optimized to minimize the ELBO of the training using the Adam optimizer (Kingma & Ba 2015), with a learning rate of $10^{-6}$, minibatch size of 32 and momentum parameters $(\beta_1, \beta_2) = (0.5, 0.999)$. The mean ELBO for the images in the validation set is monitored during training. While this decreases at first, it inevitably increases as the VAE starts overfitting the training data. We terminate training at this point (~450 epochs in practice). The test-set images are not seen by the VAE during training.

### 2.3 Validation tests

To assess the quality of our parametric model $\mu_e(z)$ for galaxy images, we present the reconstructions of galaxies from the test set that the model has not seen during training in Fig. 2. We can observe that the reconstructions of our VAE model are denoised versions of the original images that reliably contain galaxy substructure. This indicates the latent space learned by the VAE contains a point corresponding to each of the galaxies, even though they have complex and varied morphologies.

It is also important that the VAE’s reconstructions are equivariant under transformations such as rotations, since the parameter inferences by the analysis pipeline should be as well. In Fig. 3, a galaxy image from the test set is rotated by various angles. Each of the
The geometry of the system is displayed where $\alpha$ taking the mean of the decoded distribution $\hat{x}$ from the encoded distribution, and taking the mean of the decoded distribution $x = \mu(x)$.

In the source plane $\beta$ encoder, sampling reconstructions were obtained by passing the observations $x$ through the encoder, sampling $x \sim \mathcal{N}(\mu_e(x), \sigma_e(x))$ from the encoded distribution, and taking the mean of the decoded distribution $\bar{x} = \mu_e(\bar{x})$.

Reconstructions of galaxies from the test set. The input images are shown in the top row and the reconstructions in the bottom row. The reconstructions were obtained by passing the observations $x$ through the encoder, sampling $x \sim \mathcal{N}(\mu_e(x), \sigma_e(x))$ from the encoded distribution, and taking the mean of the decoded distribution $\bar{x} = \mu_e(\bar{x})$.

Figure 2. Reconstructions of galaxies from the test set. The input images are shown in the top row and the reconstructions in the bottom row. The reconstructions were obtained by passing the observations $x$ through the encoder, sampling $x \sim \mathcal{N}(\mu_e(x), \sigma_e(x))$ from the encoded distribution, and taking the mean of the decoded distribution $\bar{x} = \mu_e(\bar{x})$.

Figure 3. Approximate rotational invariance of reconstructions. The first row shows a galaxy image from the test set and versions rotated counter-clockwise by the amount indicated above each column, with new pixels filled using the known noise distribution. The second row shows the corresponding VAE reconstructions, with derotated versions shown in the third row to make comparison simpler. The same colour scale is used in each subplot.

rotated images is then derotated for comparison. The reconstructed images are once again denoised versions of the input images, and it can be seen by inspection that the derotated reconstructions are nearly identical. From this, we conclude that our generative model has (approximately) learned rotational equivariance, even though the training data set was not augmented with rotated images to teach this explicitly.

3 STRONG LENSING

Here, we review the physics of strong gravitational lensing, describe how we model the main lens and external shear, and explain how we construct the source model using the VAE from the previous section.

3.1 Strong lensing physics

A gravitational galaxy–galaxy lensing system mainly consists of a background galaxy, playing the role of the source, and a foreground galaxy, acting as the main lens that deflects the light through its gravitational potential. In the thin lens approximation, the relation between the 2D angular coordinates of the lens plane $\theta$ and the ones in the source plane $\alpha$ is encoded by the lens equation (Kormann et al. 1994; Treu 2010)

$$\alpha = \theta - \alpha(\theta),$$

where $\alpha$ is the displacement field, which defines the deflection experienced by the light ray. The geometry of the system is displayed in Fig. 4. The displacement field depends on the Newtonian gravitational potential related to the mass distribution of the foreground galaxy. By means of the Poisson equation, it can be expressed as

$$\alpha = \frac{4}{c^2} \frac{G_N}{D_{OL} D_{LS}} \int \frac{\Sigma(\theta')}{|\theta - \theta'|^2} \, d^2 \theta',$$

where $\Sigma$ is the surface mass density of the lens, and the quantities $D_{OL}, D_{OS},$ and $D_{LS}$ are the angular diameter distances between the observer and the lens, the observer and the source, and the lens and the source, respectively. Moreover, $G_N$ is the gravitational constant, while $c$ is the speed of light.

Since the lens equation (equation 6) preserves the surface brightness (photon flux density per unit angular area), the image of the system in the lens plane, denoted as $\mathcal{I}_{\text{lens}}$, is simply obtained by evaluating the source light distribution $\mathcal{I}_{\text{src}}$ on the lens plane. Hence, we have

$$\mathcal{I}_{\text{lens}}(\theta) = \mathcal{I}_{\text{src}}(\theta - \alpha(\theta)).$$

This equation is solved on a square pixel grid defined in the lens plane according to the observed image. In particular, we consider a grid of $256 \times 256$ pixels with angular size 10 arcsec $\times$ 10 arcsec, corresponding to a pixel size of 0.04 arcsec. Then, the predicted lensed image $\mathcal{I}_{\text{pred}}$ is obtained by taking into account the point spread function (PSF), which defines how a point-like source (a pixel) is spread due to atmospheric distortions and defects in the optics, and the noise from instrumental and astrophysical backgrounds. In this paper, we consider a symmetric 2D Gaussian PSF with a standard deviation of 0.05 arcsec. Moreover, to each pixel we add uncorrelated Gaussian noise with $\sigma_{\text{noise}} = 0.333, 0.1, \text{and } 0.0333$ for mock images with low, medium, and high S/N ratio, respectively. Hence, we have

$$\mathcal{I}_{\text{pred}} = \mathcal{N}(\text{PSF} * \mathcal{I}_{\text{lens}}, \sigma_{\text{noise}}),$$

where the symbol $*$ stands for the mathematical convolution between the PSF and the image in the lens plane.

The two main ingredients describing a gravitational lensing system are, therefore, the total mass distribution of the lens $\Sigma$ (the lens model) and the surface brightness profile of the background source $\mathcal{I}_{\text{src}}$ (the source model).

3Note that we do not consider the light distribution of the foreground galaxy since it is in general subtracted in real data analyses.

4For reference, this is the pixel size of the optical/UV CCDs of WFC3 Hubble (Dressel 2012).
3.2 Lens model

In a typical lensing system, the dominant contributions to the total displacement field \( \alpha \) come from the smooth main halo (mh) of the foreground galaxy and the external shear (ext), namely

\[
\alpha = \alpha_{\text{mh}} + \alpha_{\text{ext}}. \tag{10}
\]

For the first component, we consider the so-called singular power-law ellipsoid (SPLE) profile to model the surface mass distribution of the main halo. Such profiles can fit gravitational potentials of lenses in images of galaxy-scale strong lensing systems at the percent level (see e.g. Suyu et al. 2009). In this case, we have (Kassiola & Kovner 1993; Barkana 1998)

\[
\Sigma_{\text{mh}} = \Sigma_{\text{cr}} \frac{3 - \gamma}{2} \left( \frac{\rho(\theta', q)}{r_{\text{Ein}}} \right)^{1-\gamma}, \tag{11}
\]

where \( \Sigma_{\text{cr}} = c^2 D_{\text{OS}}/(4 \pi G D_{\text{OS}} D_{\text{LS}}) \) is the critical surface mass density, \( \gamma \) is the slope, \( r_{\text{Ein}} \) denotes the Einstein radius, and \( \rho \) encodes the dependence on the position. In the special case \( \gamma = 2 \), this distribution reduces to the one for a singular isothermal ellipsoid (Kormann et al. 1994). In the coordinates system \( \theta' = (\theta_2 - \theta_1) \), \( \phi \) with the galaxy is located. The corresponding displacement field \( \alpha_{\text{mh}} \) induced by the main halo mass distribution is simply given by plugging equation (11) into equation (7).

Since numerical integration is difficult to implement in an automatically differentiable manner, we instead compute \( \alpha_{\text{mh}} \) by interpolating over a pre-computed grid, as described in detail in Appendix A. This enables computation of gradients of \( \alpha_{\text{mh}} \) since the interpolation function is itself automatically differentiable.

The external shear contribution represents the additional angular structure provided by additional matter distribution in the cluster where the galaxy is located. The corresponding displacement field is parametrized as

\[
\alpha_{\text{ext}} = \left( \frac{\gamma_1}{\gamma_2} \right) \theta. \tag{13}
\]

Our lens model is thus completely defined by a set of eight parameters: \( \Theta_{\text{lens}} \equiv \{ \gamma_1, \gamma_2, \phi, q, \gamma, r_{\text{Ein}}, \theta_{\text{lens,1}}, \theta_{\text{lens,2}} \} \).

3.3 Source model

The VAE’s decoder \( d_\theta(z) \) provides a parametrized model for \( 64 \times 64 \)-pixel galaxy images, which is the basis for our source model. One of its parameters is the 64D vector \( z \) specifying a point in the VAE’s latent space. As described earlier, the prior for \( z \) is defined by fitting a multivariate normal distribution to the means of the encoded distributions of the training points and increasing its covariance by a factor of 9. We introduce four other parameters to complete the model: \( \theta_{\text{src,1}}, \theta_{\text{src,2}}, s, \) and \( i \). The first two are the position of the centre of the decoded image \( d(z) \). The second specifies the spatial scale of the image and the third is the normalization of the pixel intensities. In this work, we adopt the priors

\[
\theta_{\text{src,1}}, \theta_{\text{src,2}} \sim \mathcal{N}(0, 0.1) \tag{14}
\]

s \sim \mathcal{N}(5, 1) \tag{15}

\[
i \sim \mathcal{N}(1, 0.5) \tag{16}
\]

for our mock data generation and analysis. Our overall model for the surface brightness of the source galaxy at a position \( \theta = (\theta_1, \theta_2) \) is thus

\[
I_{\text{src}}(\theta|\Theta_{\text{src}}) = i \cdot d(z) \left( \frac{\theta_1 - \theta_{\text{src,1}}}{s}, \frac{\theta_2 - \theta_{\text{src,2}}}{s} \right), \tag{17}
\]

where \( \Theta_{\text{src}} \equiv \{ z, \theta_{\text{src,1}}, \theta_{\text{src,2}}, s, i \} \). We use bilinear interpolation to allow the right-hand side of this expression to be evaluated between adjacent pixels in the \( 64 \times 64 \) output image from the decoder.

4 THE LENSING PIPELINE

Our lensing pipeline is unique since it combines the VAE-based source and physical lens models detailed in the previous section in a fully differentiable manner. As sketched in Fig. 5, the pipeline consists of a forward flow (black solid lines) and a backward one (red dashed lines).

In the forward flow, the predicted lensed image \( I_{\text{pred}}(\Theta_{\text{lens}}, \Theta_{\text{src}}) \) is obtained once the displacement field \( \alpha(\theta|\Theta_{\text{lens}}) \) and the source surface brightness \( I_{\text{src}}(\theta|\Theta_{\text{src}}) \) have been computed in the lens model (see Section 3.2) and the source model (see Section 3.3), respectively. This image is then compared with the observed one to estimate the likelihood function given the parameters of the lens and the source models. Thanks to the differentiable programming framework, it is then possible to compute the derivatives of the likelihood function with respect to all the parameters, \( \Theta_{\text{lens}} \) and \( \Theta_{\text{src}} \). This step represents the backward flow of the whole pipeline.

The pipeline is implemented in a differentiable probabilistic programming framework comprised of the PYTORCH (Paszke et al.
machine learning library and PYRO (Bingham et al. 2018) probabilistic programming library. PYTORCH provides an AD engine, enabling the backward flow of the pipeline. The likelihood calculations in the forward flow are made straightforward by PYRO. The VAE source model is constructed from neural network layers contained in PYTORCH and trained using the variational inference module in PYRO. We employ PYRO’s variational inference and HMC modules for parameter fitting and posterior sampling our mock data analysis in the following section.

We stress that automatic differentiability is automatically guaranteed if all the pipeline is written in the differentiable programming language. Moreover, we note that the lensing pipeline is implemented so that the lens and the source models are independent building blocks. In this paper, the former is fully based on physical models while the latter is provided by the VAE’s decoder. However, thanks to the modularity of the pipeline, both can be easily modified or substituted, as can any of the priors on the lensing parameters. This fundamental feature allows one to generalize the present lensing pipeline to analyse more realistic systems and to include the gravitational effect of DM substructures in the lensing physics. This is left for future investigation.

5 RESULTS

In this section, we test the lensing pipeline on mock lensing system observations. We describe the generation of mock data, and discuss the results obtained by the parameter fitting and the posterior analysis for two different mock lensing systems.

5.1 Mock data

To test our pipeline, we generate mock lensing system observations. We create mock sources by first denoising images from the test data set using the non-local means algorithm (Buades, Coll & Morel 2005, 2011). These images are then rescaled to fill roughly the central third of the source image plane, which ensures the lensed image pixel intensities drop to zero along the image boundaries. The mock lensing parameters are determined randomly by drawing from the following distributions:

\[ \gamma_1, \gamma_2 \sim \mathcal{N}(0, 1) \]
\[ \phi \sim \mathcal{N}(0, 1 \text{ rad}) \]
\[ q \sim \mathcal{N}(0.5, 0.5) \]
\[ r_{\text{lin}} \sim \mathcal{N}(1.5 \text{ arcsec}, 0.5 \text{ arcsec}) \]
\[ \gamma \sim \mathcal{N}(2, 0.5) \]
\[ \theta_{\text{len}1}, \theta_{\text{len2}} \sim \mathcal{N}(0, 0.5). \]

The mock lensed images are then produced using the formalism from the previous section on a 256 × 256 pixel grid with angular size 10 arcsec × 10 arcsec. Finally, we convolve with the PSF and add Gaussian pixel noise.

To test how data quality impacts our pipeline’s performance, we fix the pixel noise level \( \sigma_{\text{noise}} \) to three different values to obtain observations with low, medium, and high S/N ratios. This ratio is defined by creating a mask \( m \) to select only the pixels belonging to the galaxy in the observed image \( I_{\text{obs}} \) and computing the following quantity (see e.g. O’Riordan, Warren & Mortlock (2019)):

\[ \frac{S/N(\sigma_{\text{noise}})}{\sigma_{\text{noise}}} = \frac{\sum_{i,j} m_{ij} I_{\text{obs}}(i,j)}{\sigma_{\text{noise}} \sqrt{\sum_{i,j} m_{ij}}}. \]

The mask is constructed by first convolving \( I_{\text{obs}} \) with a Gaussian \( G \) with standard deviation 0.16 arcsec (four pixels) and then thresholding the blurred image using the noise level:

\[ m_{ij} = \begin{cases} 1 & (G * I_{\text{obs}})(i,j) \geq \sigma_{\text{noise}} \\ 0 & (G * I_{\text{obs}})(i,j) < \sigma_{\text{noise}} \end{cases}. \]

Changing the threshold and/or width of the Gaussian \( G \) does not substantially change the value of S/N. The pixel noise levels and corresponding S/N values for each system are shown in Table 1 for completeness, though their specific values are not important for our study.

In the rest of this paper, we focus on two particular mock systems, hereafter referred to as system 1 and system 2. Source 1 has a fairly simple spiral morphology. Source 2 has a significant amount of structure, making it representative of complex, high-redshift source galaxies.

5.2 Parameter fitting

We first test our pipeline by finding the best-fitting source and lens parameter values. For a given image \( I_{\text{obs}} \), we compute the MAP parameter estimates:

\[ \Theta_{\text{src}}, \Theta_{\text{len}} = \max_{\Theta_{\text{len}}, \Theta_{\text{src}}} p(\Theta_{\text{len}}, \Theta_{\text{src}} | I_{\text{obs}}), \]

\[ p(\Theta_{\text{len}}, \Theta_{\text{src}} | I_{\text{obs}}) \propto p(I_{\text{obs}} | \Theta_{\text{len}}, \Theta_{\text{src}}) p(\Theta_{\text{len}}) p(\Theta_{\text{src}}). \]

The first term on the right-hand side of the posterior is the likelihood of the observed image for fixed source and lens parameters, which is Gaussian due to our noise assumption:

\[ \sigma_{\text{noise}}. \]

The second term on the prior on the lens parameters, for which we adopt equations (18)–(23). The source priors are specified in equations (14)–(16), and the prior on \( z \) is described in Section 2.

Since it is possible to differentiate through the full lensing pipeline, we obtain the best-fitting parameters using the gradient-based Adam optimizer (Kingma & Ba 2015). Adam has been empirically shown to outperform other gradient-based methods in non-convex optimization problems with large numbers of parameters. These fits converge after \( \sim 10^3 \) iterations, which takes approximately 20 (7.5) min on CPU (GPU).

The MAP reconstructions of the mock lensing systems 1 and 2 are shown in Figs 6 and 7. We report the true and fit images, and the residuals between these, in the lens and in the source planes, for three values of S/N ratio. In case of system 1, the residuals in the lens plane are at the noise level even for the smallest pixel noise.
Figure 6. Results of MAP fit of mock lensing system 1. The different columns show the true and fit observations, the residual between these, the true and fit sources and the residuals between those. The rows correspond to different observed S/N values. Within each row, the colour scales for the true and fit observations are the same, as are the scales for the true and fit sources. The observation residuals are normalized by dividing by the standard deviation of the observation noise. The source residuals are normalized by dividing by 10 per cent of the maximum value of the source. The colour scale ranges from dark blue to bright yellow.

Figure 7. Results of MAP fit of mock lensing system 2. The subplots and scales are explained in the caption of Fig. 6.

level (highest S/N) considered. This is related to the fact that the reconstruction of the source improves as the S/N ratio increases, as can clearly seen in the last column in Fig. 6. This does not occur for the complex source galaxy of system 2. Indeed, as shown in Fig. 7, even for the highest S/N value, the VAE’s decoder is not able to reproduce all the substructures exhibited in the source surface brightness. This directly affects the image reconstruction in the lens plane, and the corresponding residuals (third column in Fig. 7) increase and show more structure as the S/N ratio increases.

5.3 Posterior analysis

To study our pipeline’s parameter inference capabilities, we ran HMC (Duane et al. 1987; Neal 2012; Betancourt 2017) to sample from the parameters’ posteriors. HMC is an MCMC procedure that uses Hamiltonian dynamics based on the gradient of the posterior to efficiently traverse parameter space. HMC can take larger steps than other MCMC procedures such as Metropolis–Hastings while keeping the acceptance probabilities high, leading
Differentiable strong lensing

6 CONCLUSIONS

We have presented the first step towards a new inference pipeline to analyse present and future strong gravitational lensing systems. The main novelty of our approach is the use of a differentiable probabilistic programming framework in which all operations are automatically differentiable with respect to the input parameters. This powerful approach makes Bayesian inference feasible for complex models with hundreds (or thousands) of free parameters thanks to efficient sampling techniques utilizing gradient descent.

Our lensing pipeline, shown in Fig. 5, consists of two independent blocks describing the surface brightness of the source galaxy (source model) and the mass distribution of the lens galaxy (lens model). They are combined to generate the lensed image, which is used to estimate the likelihood and perform Bayesian inference. The advantages of this strategy are:

(i) Exact gradients of the pipeline’s output can be computed with respect to its inputs using AD. This makes it possible to use efficient gradient-based fitting and posterior sampling procedures.

(ii) Using a differentiable probabilistic programming framework allows us to integrate the VAE source model learned from unlensed galaxy images directly with a physical lensing model. Learning the source model directly from data removes the need for tuning hyperparameters to regularize the source model.

(iii) We fully automatize the inference step using probabilistic programming. The lens and source parameters are fit and sampled simultaneously and can have arbitrary priors.

(iv) By implementing our pipeline in the PYTORCH framework, we automatically gain the ability to perform computations on GPUs.

From a quantitative perspective, the best-fitting lens parameter values we obtained in our mock data tests were within $\sim 1\%$ of the true values, albeit with some bias at very high S/N ratios.

Since our fitting and posterior sampling is gradient-based, the computational cost of adding parameters to increase the realism of our model is low. For example, a prerequisite for applying our pipeline to real data is to model the light from the lens galaxy. We anticipate this could be done by adding an additional unlensed light source again modelled by our VAE. In future work, we will also explore more realistic lens models with additional components such as DM subhaloes, a baryonic disc, line-of-sight haloes, or a more complicated main lens model. For these very high-dimensional models, automated gradient-based inference
techniques with favourable scaling behaviour such as variational inference (Hoffman et al. 2012; Blei et al. 2016) could enable analysis of posterior distributions for hundreds or even thousands of parameters. Lastly, we expect that our pipeline’s efficiency in analysing large data sets could be improved by making technical changes so it can operate on batches of images in parallel, as is done when training CNNs.

While our pipeline is an early example of differentiable probabilistic programming, we anticipate this approach will enable other challenging and exciting data analyses in the future by leveraging the advantages of deep learning and physics modelling.

ACKNOWLEDGEMENTS

We would like to thank Simona Vegetti and Rajat Thomas for helpful conversations. This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative. We acknowledge funding from the Netherlands Organization for Scientific Research (NWO) through the VIDI research program ‘Probing the Genesis of Dark Matter’ (680-47-532).

REFERENCES

Birrer S., Amara A., 2018, Physics of the Dark Universe, 22, 189

M. Chianese et al.
In this appendix, we describe how the contribution to the displacement field due to the main halo, the SPLE profile, is implemented in the lens model block of the lensing pipeline shown in Fig. 5. It is worth observing that, in the absence of an analytical expression for $\alpha_{0\text{th}}$, one has to numerically compute the integral in equation (7). However, this is not feasible in our framework because the numerical integration is not coded up in a autodifferentiable way. For this reason, the displacement field is instead determined by means of an interpolation of a precomputed numerical table of the corresponding integral, equation (7).

In case of a surface mass density profile with elliptical contours (like e.g. the SPLE profile), the 2D integral of equation (7) can be reduce to a simpler 1D integral (Schramm 1990; Barkana 1998)

\[
\alpha_{1}(\theta_1, \theta_2) = 2\theta_1 \int_0^{\rho(\theta_1, \theta_2)} \frac{\rho^2(\rho') \omega}{\theta_1^2 + \alpha_1^2 \theta_1^2} d\rho',
\]

(A1)

\[
\alpha_{2}(\theta_1, \theta_2) = 2\theta_2 \int_0^{\rho(\theta_1, \theta_2)} \frac{\rho^2(\rho') \omega}{\theta_2^2 + \alpha_2^2 \theta_2^2} d\rho',
\]

(A2)

with

\[
\omega^2 = \frac{\Delta + r^2 + \rho^2(1 - q^2)}{\Delta + r^2 - \rho^2(1 - q^2)},
\]

(A3)

\[
\Delta^2 = \rho^2(1 - q^2) + \theta_1^2 - \theta_2^2 - 4\theta_1^2 \theta_2^2.
\]

(A4)

where $r = \theta_1^2 + \theta_2^2$, $\rho(\theta_1, \theta_2)$ is defined in equation (13), and the probability $\kappa$ is the surface mass density $\Sigma$ in units of the critical density $\Sigma_{\text{cr}}$. For a given profile, the integrals in equations (A1) and (A2) can be tabulated for different values of a subset of the lens parameters $\Theta_{\text{tan}}$. In particular, in case of the SPLE lens, the displacement field has been evaluated on a unit circle ($r = 1$), for which the coordinates are $\theta_1 = \cos \eta$ and $\theta_2 = \sin \eta$ with the angle $\eta$ being in the first quadrant ($0 \leq \eta \leq \pi/2$), for different values of the axial ratio $q$ in the interval [0, 1] and the slope $\gamma$. The Einstein radius is instead fixed to be $\theta_{\text{Ein}} = 1$. Such a procedure provided a 3D table in the variables $\{q, \theta, \gamma\}$. This numerical table is then interpolated to compute the SPLE displacement field for any values of $q$ and $\gamma$, and at any position $(\theta_1, \theta_2)$. Each component of the displacement field is indeed given by

\[
\alpha_{1}(\theta_1, \theta_2) = \alpha_{1}(q, \theta_2) \text{sign}(\theta_1),
\]

(A5)

\[
\alpha_{2}(\theta_1, \theta_2) = r^{-\gamma} \alpha_{2}(\theta_1, \theta_2)_{r_{\text{Ein}} = 1}.
\]

(A6)

Moreover, in case of the SPLE profile, these two components show a simple scaling relation as a function of the Einstein radius. We have

\[
\alpha_{1}(\theta_1, \theta_2)_{r_{\text{Ein}}} = \left(\frac{1}{r_{\text{Ein}}} \right)^{1 - \gamma} \alpha_{1}(\theta_1, \theta_2)_{r_{\text{Ein}} = 1}.
\]

(A7)
It is worth noticing that the accuracy in computing the displacement field by means of this procedure depends on the size of the interpolation table, which can be defined without any constraint.

APPENDIX B: ELBO DERIVATION

This appendix demonstrates one possible derivation of equation (4). Consider a latent variable model defined by the joint probability distribution \( p(x, z) \), where \( x \) and \( z \) are the observed and latent variables, respectively. We start by rewriting an expression for the log of the evidence:

\[
\log p(x) = \log \int_z p(x, z) dz = \log \int_z p(z|x) \frac{p(x, z)}{p(z|x)} dz, \tag{B1}
\]

This can be recognized as an expectation value over \( p(z|x) \):

\[
\log p(x) = \mathbb{E}_{p(z|x)} \left[ \frac{p(x, z)}{p(z|x)} \right]. \tag{B2}
\]

By Jensen’s inequality, which relates the expectation value of a convex function to that function applied to an expectation value, we have

\[
\log p(x) \geq \mathbb{E}_{p(z|x)} \left[ \log \frac{p(x, z)}{p(z|x)} \right], \tag{B3}
\]

and thus:

\[
\log p(x) \geq \mathbb{E}_{p(z|x)} \left[ \log \frac{p(x, z)}{p(z|x)} \right]. \tag{B5}
\]

Using the definition of the KL divergence for continuous random variables

\[
D_{KL}[p||q] = -\mathbb{E}_{p(z|x)} \left[ \log \frac{q(z|x)}{p(z|x)} \right], \tag{B6}
\]

this can be manipulated to yield

\[
\log p(x) \geq \mathbb{E}_{p(z|x)} \left[ \log \frac{p(x, z)}{p(z|x)} \right] = \mathbb{E}_{p(z|x)} \left[ \log p(x|z) + \log \frac{p(z)}{p(z|x)} \right] \tag{B7}
\]

\[
= \mathbb{E}_{p(z|x)} \left[ \log p(x|z) \right] - D_{KL}[p(z|x)||p(z)] \tag{B8}
\]

\[
= \text{ELBO}(\theta, \phi; x). \tag{B10}
\]

The VAE training objective is obtained by substituting the approximate distributions \( p(x|z) \rightarrow d_\theta(x|z) \) and \( p(z|x) \rightarrow e_\phi(z|x) \) for the true ones.

APPENDIX C: OPTIMIZING THE ELBO

Training the VAE requires taking the gradient of the ELBO for (batches of) training images \( \{x^{(i)}\}_{i=1}^N \) with respect to the encoder and decoder’s parameters \( \theta \) and \( \phi \):

\[
\nabla_{\theta, \phi} \text{ELBO}(\theta, \phi; x^{(i)}) = \nabla_{\theta, \phi} \mathbb{E}_{e_\phi(z|x^{(i)})} \log d_\theta(x^{(i)}|z)] - \nabla_{\theta, \phi} D_{KL}[e_\phi(z|x^{(i)})||p(z)]. \tag{C1}
\]

For the normal encoding distribution and latent space prior adopted in this work, the KL divergence term can be integrated analytically, which makes it simple to compute the second term on above. The first term is more challenging: while a Monte Carlo estimate of the derivative with respect to \( \theta \) can be performed by sampling \( \{z^{(j)} \sim e_\phi(z|x^{(i)})\}_{j=1}^M \), it is not obvious how to compute the derivatives of the sampled latent variable values with respect to \( \phi \).

The solution is the reparametrization trick introduced in the two original papers on VAEs (Kingma & Welling 2013; Jimenez Rezende et al. 2014). The insight is that (assuming a normal encoding distribution) the randomness and \( \phi \)-dependent parts of the sampling process can be factored, allowing the sampled values to be written as

\[
z^{(j)} = \mu_e(x^{(i)}, \phi) + \epsilon^{(j)} \sigma_e(x^{(i)}; \phi), \tag{C2}
\]

with \( \epsilon^{(j)} \sim \mathcal{N}(0, I) \). These can be used to construct the following Monte Carlo gradient estimator by treating the \( \epsilon^{(j)} \) values as constants for the optimization epoch:

\[
\nabla_{\theta, \phi} \mathbb{E}_{e_\phi(z|x^{(i)})} \log d_\theta(x^{(i)}|z^{(j)}) \approx \frac{1}{M} \sum_{j=1}^M \log d_\theta(x^{(i)}|z^{(j)}). \tag{C3}
\]

This estimator is stable; we set \( M = 10 \) in our work by using batches of 32 training images.

APPENDIX D: VARIATIONAL AUTOENCODER ARCHITECTURE

The architectural details of the encoder and decoder networks of our VAE are presented in Tables D1 and D2, respectively. All weights were initialized to 0.02. The biases in the final linear layers of the encoder were initialized to 0.

We tried several experiments to see whether we could improve upon this VAE design. For example, since the decoder can have trouble saturating the final tanh activation, we tried exchanging this for a LeakyReLU, as well as removing it altogether. We also tested 32 and 128 latent-space dimensions. The former lead to blurry reconstructions while the latter yielded little improvement relative to 64 latent-space dimension. Dai & Wipf (2019) analytically demonstrated that making the hyperparameter \( \sigma_d \) a trainable parameter should lead to sharper reconstructions. We did not find this to be the case, and instead found that \( \sigma_d \) converged to roughly the value we selected by hand based on the S/N ratio of the training data. Finally, we experimented with a residual network-based architecture, as was used in Dai & Wipf (2019), which did not yield any improvement relative to the architecture we eventually selected.
Table D1. Encoder neural network architecture. The notation uses the same conventions as PYTORCH. The arguments of Conv2d indicate the number of input channels, number of output channels, kernel size, stride, and zero padding; all convolutions are unbiased. The LeakyReLU argument is slope for inputs less than 0. The BatchNorm2d argument is the number of input channels. The output of the last convolutional block is flattened before being passed to the two separate Linear layers to produce the mean and standard deviation of the encoding distribution. Linear layers’ arguments show the number of input and output channels.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Arguments</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d</td>
<td>(1, 64, 4, 2, 1)</td>
<td>LeakyReLU(0.2)</td>
</tr>
<tr>
<td>Conv2d</td>
<td>(64, 128, 4, 2, 1)</td>
<td>BatchNorm2d(128)</td>
</tr>
<tr>
<td>Conv2d</td>
<td>(128, 256, 4, 2, 1)</td>
<td>LeakyReLU(0.2)</td>
</tr>
<tr>
<td>Conv2d</td>
<td>(256, 512, 4, 2, 1)</td>
<td>BatchNorm2d(512)</td>
</tr>
<tr>
<td>Conv2d</td>
<td>(512, 4096, 4, 1, 0)</td>
<td>LeakyReLU(0.2)</td>
</tr>
<tr>
<td>Linear</td>
<td>(4096, 64)</td>
<td>Linear(4096, 64)</td>
</tr>
</tbody>
</table>

\[ \mu(x) \quad \sigma(x) \]

Table D2. Decoder neural network architecture. The notation is described in the caption of Table D1, and is the same for Conv2d and ConvTranspose2d. The input vector $z$ is reshaped to have 64 channels and spatial dimensions equal to 1 along both axes.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Arguments</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvTranspose2d</td>
<td>(64, 512, 4, 1, 0)</td>
<td>BatchNorm2d(512)</td>
</tr>
<tr>
<td>ConvTranspose2d</td>
<td>(512, 256, 4, 2, 1)</td>
<td>BatchNorm2d(256)</td>
</tr>
<tr>
<td>ConvTranspose2d</td>
<td>(256, 128, 4, 2, 1)</td>
<td>BatchNorm2d(128)</td>
</tr>
<tr>
<td>ConvTranspose2d</td>
<td>(128, 64, 4, 2, 1)</td>
<td>BatchNorm2d(64)</td>
</tr>
<tr>
<td>ConvTranspose2d</td>
<td>(64, 1, 4, 2, 1)</td>
<td>Tanh</td>
</tr>
</tbody>
</table>

\[ \mu_d(z) \]

This paper has been typeset from a \TeX/\LaTeX\ file prepared by the author.