Efficient posterior inference & generalization in physics-based Bayesian inference with conditional GANs

Deep Ray
University of Southern California
Los Angeles, CA 90089, USA
deepray@usc.edu

Dhruv V. Patel
University of Southern California
Los Angeles, CA 90089, USA
dhruvvpa@usc.edu

Harisankar Ramaswamy
University of Southern California
Los Angeles, CA 90089, USA
hramaswa@usc.edu

Assad A. Oberai
University of Southern California
Los Angeles, CA 90089, USA
aoberai@usc.edu

Abstract

In this work, we propose a conditional generative adversarial network (cGAN) to sample from the posterior of physics-based Bayesian inference problems. We utilize a U-Net architecture for the generator and inject the latent variable using conditional instance normalization. We solve the inverse heat conduction problem and demonstrate how the proposed strategy effectively quantifies the uncertainty in the inferred field. We also show that the structure of the generator promotes generalizability due to the local-nature of the learned inverse map.

1 Introduction

Inverse problems are ubiquitous in science and engineering in areas such as computerized tomography [1], seismology [2, 3], climate-modeling [4, 5], and astronomy [6]. Unlike the forward/direct problem, the inverse problem can be challenging to solve, and might lack well-possedness [7]. Bayesian inference provides a meaningful strategy to overcome these challenges by posing the problem in a stochastic framework. It combines prior knowledge and data likelihood to formulate an expression for the posterior distribution. Knowing the posterior is useful in evaluating statistical estimates and quantifying the reliability of the inferred field for a given measurement.

Over the last few years, several deep learning-based strategies have been developed to solve inverse problems. Popular among these are generative adversarial networks (GANs), which are useful in approximating the underlying distribution characterized by data [8, 9, 10, 11, 12]. In [13], a conditional generative adversarial network (cGAN) was developed to solve large-scale medical imaging inverse problems in the Bayesian framework. In particular, a U-Net architecture with convolution-residual blocks was used to construct the GAN generator. Such cGAN models have been recently extended to solve nearshore bathymetry problems in [14]. Additionally, a GAN-based approach was proposed in [15] to approximate many-to-many image mappings arising in computer vision. A U-Net architecture was used for the generator and the latent information was injected at various levels of the U-Net using conditional instance normalization [16].

In our work, we combine the two distinct but related themes from [13] and [15]. We design cGANs to learn the posterior in physics-based Bayesian problems, where the generator uses the U-Net architecture and conditional instance normalization. We consider the inverse heat conduction, and use the cGAN to infer the thermal conductivity for the steady-state problem, and the initial condition

NeurIPS 2021 Workshop on Deep Learning and Inverse Problems, virtual.

for the unsteady problem. Additionally, we demonstrate how the architecture of the generator plays a crucial role in promoting generalizability for the proposed approach.

2 Problem formulation

Consider the forward/direct problem given by $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, where $\boldsymbol{x} \in \Omega_X$ is the input while $\boldsymbol{y} \in \Omega_Y$ is the response. A typical example is determining the temperature field, given the heat source, boundary conditions and thermal conductivity. In this case, $\boldsymbol{f}$ is the solution map for the underlying PDE. The associated inverse problem is to infer the heat source or the thermal conductivity field, given the noisy temperature field; that is, inferring $\boldsymbol{x}$ given $\boldsymbol{y}$. We solve the inverse problem in a Bayesian setting, by assuming that the inferred field is sampled according to the distribution $\boldsymbol{x} \sim p_{\boldsymbol{x}}$. Then the measurements are sampled using the distribution $\boldsymbol{y} \sim p_{\boldsymbol{y}}$ induced by the forward map $\boldsymbol{f}$. We are interested in generating samples from the posterior $p_{\boldsymbol{x}|\boldsymbol{y}}$ for a given the measurement $\boldsymbol{y}$. Following the strategy proposed in [13], we train a conditional Wassesrtein GAN to achieve this goal.

We assume access to M realizations $\{\boldsymbol{x}^{(i)}\}_{i=1}^{M}$ from $p_{\boldsymbol{x}}$, and construct the dataset of pairwise samples $\mathcal{S} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{M}$ by solving the forward model, exactly or approximately. Note that the samples in $\mathcal{S}$ represent realizations from the joint distribution $p_{\boldsymbol{xy}}$. Let $\boldsymbol{z} \in \Omega_Z \subset \mathbb{R}^{N_Z}$, be the latent variable of the GAN distributed according to a distribution $p_{\boldsymbol{z}}$ which is easy to sample from. We define the GAN generator as $\boldsymbol{g} : \Omega_Z \times \Omega_Y \mapsto \Omega_X$, which generates samples from the induced distribution $p^{\text{g}}_{\boldsymbol{x}}$. Further, for a given $\boldsymbol{y}$, we denote the induced posterior distribution as $p^{\text{g}}_{\boldsymbol{x}|\boldsymbol{y}}$. The critic $d : \Omega_X \times \Omega_Y \mapsto \mathbb{R}$, is constructed to distinguish between true samples $(\boldsymbol{x}, \boldsymbol{y}) \sim p_{\boldsymbol{xy}}$ and fake samples $(\boldsymbol{x}^g, \boldsymbol{y})$, where $\boldsymbol{x}^g \sim p^{\text{g}}_{\boldsymbol{x}|\boldsymbol{y}}$. The loss function of the WGAN is given by

$$\mathcal{L}(d, \boldsymbol{g}) \equiv \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},\boldsymbol{y})\sim p_{\boldsymbol{xy}} \\ \boldsymbol{z}\sim p_{\boldsymbol{z}}}} \left[d(\boldsymbol{x}, \boldsymbol{y}) - d(\boldsymbol{g}(\boldsymbol{z}, \boldsymbol{y}), \boldsymbol{y})\right] = \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim p_{\boldsymbol{x}|\boldsymbol{y}} \\ \boldsymbol{y}\sim p_{\boldsymbol{y}}}} \left[d(\boldsymbol{x}, \boldsymbol{y})\right] - \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}^g\sim p^{\text{g}}_{\boldsymbol{x}|\boldsymbol{y}} \\ \boldsymbol{y}\sim p_{\boldsymbol{y}}}} \left[d(\boldsymbol{x}^g, \boldsymbol{y})\right]. \tag{1}$$

In the equation above, we begin with the definition of the loss term (also how it is constructed in practise), and derive the equality using the definition of the joint distribution and that of $p^{\text{g}}_{\boldsymbol{x}}$. This relation is used to derive the weak equivalence between $p_{\boldsymbol{x}|\boldsymbol{y}}$ and $p^{\text{g}}_{\boldsymbol{x}|\boldsymbol{y}}$ below. The WGAN is trained by solving the following min-max problem

$$(d^*, \boldsymbol{g}^*) = \arg\min_{\boldsymbol{g}} \arg\max_{d} \mathcal{L}(d, \boldsymbol{g}), \tag{2}$$

which is equivalent to minimizing the Wasserstein-1 distance between $p_{\boldsymbol{x}|\boldsymbol{y}}$ and $p^{\text{g}}_{\boldsymbol{x}|\boldsymbol{y}}$ under the assumption that d is 1-Lipschitz [13]. The Lipschitz constraint on the critic can be weakly imposed using a gradient penalty term [17] while training d. Further, convergence in the Wasserstein-1 metric is equivalent to the weak convergence of the two measures, i.e.,

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{\boldsymbol{x}|\boldsymbol{y}}} \left[l(\boldsymbol{x})\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}^g\sim p^{\text{g}}_{\boldsymbol{x}|\boldsymbol{y}}} \left[l(\boldsymbol{x}^g)\right] = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}} \left[l(\boldsymbol{g}^*(\boldsymbol{z}, \boldsymbol{y}))\right], \quad \forall\, l \in C_b(\Omega_X). \tag{3}$$

This implies that for a given measurement $\boldsymbol{y}$, any statistic of the posterior distribution can be evaluated by sampling $\boldsymbol{z}$ from $p_{\boldsymbol{z}}$ and pushing it through the trained generator $\boldsymbol{g}^*(\cdot, \boldsymbol{y})$.

The proposed algorithm is: (a) Generate or collect $\{\boldsymbol{x}^{(i)}\}_{i=1}^{M}$ sampled from $p_{\boldsymbol{x}}$. (b) For each $\boldsymbol{x}^{(i)}$ solve the forward model to determine $\boldsymbol{y}^{(i)}$. (c) Train the WGAN using the set of pairwise samples $\mathcal{S} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{M}$. (d) For a given measurement $\boldsymbol{y}$, compute the desired statistics of the posterior by sampling $\boldsymbol{z}$ from $p_{\boldsymbol{z}}$ and pushing it through the trained generator $\boldsymbol{g}^*(\cdot, \boldsymbol{y})$.

3 Numerical results

We consider the two-dimensional heat conduction problem. Given a noisy measurement of the temperature field, we infer the thermal conductivity field for the steady-state problem, and the initial condition for the unsteady problem (see Appendix B). The cGANs comprise of generators with a U-Net architecture. The latent variable $\boldsymbol{z}$ is injected at various levels of the U-Net using conditional instance normalization, instead of concatenating it with the input $\boldsymbol{y}$. This approach gives us the flexibility of choosing the latent dimension N_Z independent of the dimension of $\boldsymbol{y}$. Details of the cGAN architectures and the corresponding hyper-parameters can be found in Appendix A. The mean and standard deviation of the inferred field (for a given $\boldsymbol{y}$) are computed using 200 samples from the the latent space.

3.1 Inferring the thermal conductivity

We solve the non-linear coefficient inversion problem for steady-state heat conduction, where we infer the conductivity field κ given a noisy measurement of the temperature field $\widetilde{u}$ (see Appendix B.1). The cGAN is trained on a dataset of 8000 samples of discrete conductivity $\boldsymbol{x}$ and measurement $\boldsymbol{y}$ pairs evaluated on a 64×64 Cartesian grid. The conductivity field is taken as random circular inclusion with a uniform background value of unity. We test the performance of the trained generator on three different datasets. The first dataset consists of circular inclusions (like the training set), the second and third sets are out-of-distribution (OOD) and consist of elliptical inclusions and two circular inlcusions, respectively. In Figure 1(a)-(f), we show the predictions of our trained generator for examples from each of these test sets. In general, we notice that the trained network is able to reliably predict the position as well as orientation of the inclusions in all the three scenarios, including the OOD examples. For samples with a single circular inclusion, the mean is very close to the target. The uncertainty in prediction increases near the corners of the square domain, as one would expect, in the SD plots of Figures 1(b) and (d). Interestingly, in Figure 1(d), it appears as if the generator is attempting to combine two circles together to approximate the elongated shape of the ellipse. In the case of two circles seen in Figures 1(e) and (f), the size of the predominant circles are predicted accurately, while the smaller circles exhibit higher standard deviation. This indicates that the generator is less certain about the placement of an additional inclusion. Overall, we observe that the generator generalizes reasonably well to the OOD datasets.

3.2 Inferring the initial condition

We consider the time-dependent heat conduction equation without a source term and a constant conductivity field (see Appendix B.2). Given a noisy temperature field at time $t = 1$, we wish to infer the initial condition. This is a severely ill-posed problem as significant information is lost via the diffusion process when moving forward in time. We denote the discretized initial and final temperature field as $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, evaluated on a 28×28 Cartesian grid. The training dataset is generated using linearly scaled MNIST [18] handwritten digits as the initial condition, with the training set consisting of 8000 samples. In Figure 1(g)-(i), we present the results with the trained generator for measurements corresponding to test MNIST digits. As can be observed, the mean correctly captures the target initial condition of the samples. The SD is the highest at the boundaries of the digits, indicating a higher uncertainty in predicting the sharp transition region of initial temperature field. In Figure 1(j)-(l), we show the results when the the target initial condition is chosen from the notMNIST dataset (OOD dataset). We observe that the generator struggles to capture high-temperature regions closer to the boundary and the broader zones in the interior. Note that MNIST digits in the training set are more spatially centered and have narrower features. Nonetheless, the generator is able to visibly capture the underlying notMINST characters, once again indicating that it has the capacity to generalize beyond the training distribution. Further, we note that the standard deviation peaks not only at the characters' boundaries, but also in the interior regions where the generator failed to predict the higher-values of initial condition field.

3.3 Generalizability

The numerical results presented above indicate that the trained cGANs have the ability to perform well on measurements sampled from a test distribution that is out-of-distribution. To understand this, we first examine the locality of the inverse map learned by the generator. We consider the GAN generator trained in Section 3.2, and compute the gradient of the k-th pixel of the prediction $\boldsymbol{x}$ with respect to the network input $\boldsymbol{y}$. We evaluate the magnitude of gradient and average over 10 distinct samples of $\boldsymbol{y}$ and 10 realizations of $\boldsymbol{z}$, i.e.,

$$\overline{\text{grad}}_k = \frac{1}{100} \sum_{i=1}^{10} \sum_{j=1}^{10} \left| \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{y}}(\boldsymbol{z}^{(j)}, \boldsymbol{y}^{(i)}) \right|, \quad \boldsymbol{y}^{(i)} \sim p_{\boldsymbol{y}}, \quad \boldsymbol{z}^{(j)} \sim p_{\boldsymbol{z}}, \quad 1 \leq k \leq 28^2.$$

The averaged gradients for a few pixels k are shown in Figure 2. Note that the gradient for each pixel is concentrated in the neighbourhood of the corresponding pixel in $\boldsymbol{y}$. In other words, the domain of influence of k-th pixel of $\boldsymbol{x}$ is a neighbourhood of the k-th pixel of $\boldsymbol{y}$, but not all of $\boldsymbol{y}$. The locality of the generator is not unexpected, since most of the operations in the U-Net architecture (convolutions, transpose-convolutions, up-/down-sampling, etc) are local in nature. Although we do also implement

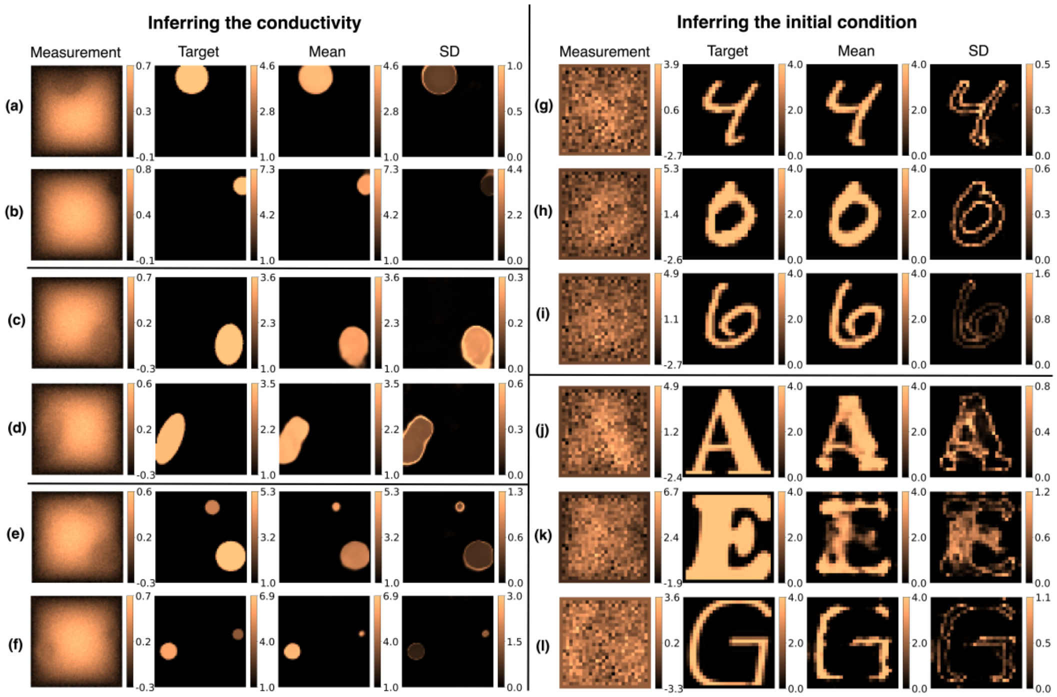

Figure 1: Inferred conductivity ((a)-(f)) and initial condition ((g)-(l)) for the heat conduction problem. (c)-(f) and (j)-(l) are results for out-of-distribution (OOD) test samples.

a conditional instance normalization in the intermediate layers, it does not seem to substantially alter the local influence $\boldsymbol{y}$ on the prediction.

Recognizing that (a) the inverse operator learned by the generator is spatially local, and (b) the GAN is trained with data (circular inclusions and MNIST digits) that contains instances of $\boldsymbol{x}$ with sharp binary contrast, we anticipate that it will generalize well to instances where the distribution of the recovered parameter consists of samples with sharp binary contrast. *This would be independent of the global form of the spatial distribution of* $\boldsymbol{x}$. This is what is observed in Figure 1, where GAN trained on instances of single circular inclusions generalizes to multiple circular inclusions and elliptical inclusion, and a GAN trained on MNIST digits generalizes to notMNIST characters.

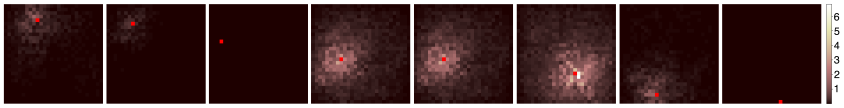

Figure 2: Average pixel-wise gradient network output (trained on MNIST) data. The red marker in each tile denotes the pixel (of $\boldsymbol{x}$) under consideration.

4 Conclusion

In this work, we have proposed training cGANs to solve physics-based Bayesian inference problems. The efficacy of this strategy has been shown by solving the inverse head conduction model in two-dimensions. The generator of the GAN has a special architecture which i) injects latent information using conditional instance normalization, giving us the flexibility to choose an arbitrary latent dimension, and ii) uses a U-Net architecture which promotes the learning of a local inverse mapping. The local-nature of the learned map promotes the generalizability of the cGAN, which is demonstrated through numerical experiments. Future work will consider additional physics-based problems and explore a mathematical explanation connecting the network's architecture and the locality of the "true" inverse map to the expected generalizability.

Acknowledgments and Disclosure of Funding

The authors acknowledge support from ARO grant W911NF2010050, NRL grant N00173-19-P-1119, and AIER (USC), and resources from the Center for Advanced Research Computing (USC).

References

[1] F. Natterer. *The Mathematics of Computerized Tomography*. Society for Industrial and Applied Mathematics, 2001.

[2] Wences P Gouveia and John A Scales. Resolution of seismic waveform inversion: Bayes versus Occam. *Inverse Problems*, 13(2):323–349, apr 1997.

[3] Öz Yilmaz. *Seismic Data Analysis: Processing, Inversion, and Interpretation of Seismic Data*. Society of Exploration Geophysicists, 01 2001.

[4] Charles Jackson, Mrinal K. Sen, Paul L. Stoffa, Charles Jackson, Mrinal K. Sen, and Paul L. Stoffa. An Efficient Stochastic Bayesian Approach to Optimal Parameter and Uncertainty Estimation for Climate Model Predictions. *Journal of Climate*, 17(14):2828–2841, jul 2004.

[5] Sixun Huang, Jie Xiang, Huadong Du, and Xiaoqun Cao. Inverse problems in atmospheric science and their application. *Journal of Physics: Conference Series*, 12:45–57, jan 2005.

[6] I. Craig and John C. Brown. Inverse problems in astronomy, a guide to inversion strategies for remotely sensed data. 1986.

[7] Jacques Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton university bulletin*, pages 49–52, 1902.

[8] Alexander Y. Sun. Discovering state-parameter mappings in subsurface models using generative adversarial networks. *Geophysical Research Letters*, 45(20):11,137–11,146, 2018.

[9] Yibo Yang and Paris Perdikaris. Adversarial uncertainty quantification in physics-informed neural networks. *Journal of Computational Physics*, 394:136–152, 2019.

[10] Liu Yang, Dongkun Zhang, and George E.M. Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *SIAM Journal on Scientific Computing*, 42(1):A292–A317, feb 2020.

[11] Dhruv V Patel and Assad A Oberai. Gan-based priors for quantifying uncertainty. *arXiv preprint arXiv:2003.12597*, 2020.

[12] Teeratorn Kadeethum, Daniel O'Malley, Jan Niklas Fuhg, Youngsoo Choi, Jonghyun Lee, Hari S. Viswanathan, and Nikolaos Bouklas. A framework for data-driven solution and parameter estimation of pdes using conditional generative adversarial networks, 2021.

[13] Jonas Adler and Ozan Öktem. Deep bayesian inversion. *arXiv preprint arXiv:1811.05910*, 2018.

[14] Yizhou Qian, Mojtaba Forghani, Jonghyun Harry Lee, Matthew Farthing, Tyler Hesser, Peter Kitanidis, and Eric Darve. Application of deep learning-based interpolation methods to nearshore bathymetry, 2020.

[15] Amjad Almahairi, Sai Rajeshwar, Alessandro Sordoni, Philip Bachman, and Aaron Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. In *International Conference on Machine Learning*, pages 195–204. PMLR, 2018.

[16] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style, 2017.

[17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[18] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

A WGAN architecture

The cGANs in this work were trained using the following loss function

$$\mathcal{L}(d,\boldsymbol{g}) = \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},\boldsymbol{y})\sim p_{\boldsymbol{xy}}\\ \boldsymbol{z}\sim p_{\boldsymbol{z}}}} \left[d(\boldsymbol{x},\boldsymbol{y}) - d(\boldsymbol{g}(\boldsymbol{z},\boldsymbol{y}),\boldsymbol{y}) + \lambda \mathop{\mathbb{E}}_{\epsilon\sim\mathcal{U}(0,1)} \left[(\|\partial_1 d(\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\epsilon),\boldsymbol{y})\|_2 - 1)^2 \right] \right], \tag{4}$$

which has been augmented with a gradient penalty term. Here, $\mathcal{U}(0,1)$ denotes the uniform distribution on $[0,1]$, $\partial_1 d(.,.)$ denotes the derivative with resespect to the first argument, and

$$\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\epsilon) = \epsilon\boldsymbol{x} + (1-\epsilon)\boldsymbol{g}(\boldsymbol{z},\boldsymbol{y}).$$

For all experiments, we set the gradient penalty parameter as $\lambda = 10$.

All networks are trained on TensorFlow. Each network took less than 20 hours to train on a single NVIDIA Tesla K40 GPU.

To describe the architecture of the generator and discriminator, we first introduce the following notations:

- `norm` denotes the type of normalization used. This can take the value `cin` for condition instance normalization using the latent variable or `ln` for layer normalization. Note that `cin` implicitly implies that the latent variable is also an input to the particular layer using the normalization.
- `Conv(k,n)` denotes a 2D convolution with `k` filters of size `n` and stride 1. If `n > 1`, reflective padding of width 1 is applied in the spatial dimensions of the input before applying the convolution.
- `LReLU` denotes the leaky ReLU activation with parameter 0.1.
- `Res(norm)` denotes a residual block whose action on an input `X` with `k` channels is given by:
  ```
  X →norm →Conv(k,1) →X1
  X →norm →LReLU →Conv(k,3) →norm →LReLU →Conv(k,3) →X2
  X1 + X2 →Block_Output
  ```
  If the `norm` argument is absent, then no normalization is applied in the residual block.
- `Down(k)` denotes a down-sampling block which applies `Conv(k,3)` and `LReLU` operations to the input, and then reduces the spatial resolution by a factor of 2.
- `CC(X_old)` denote the the concatenation of `X_old` to the input along the channel direction. This helps implement skip connections.
- `Up(k,X_old)` denotes an up-sampling block which first applies `CC( X_old)`, `Conv(k,3)` and `LReLU` operations to the input, and then increases the spatial resolution by a factor of 2 using nearest neighbour interpolation. If the argument `X_old` is absent, then no concatenation is applied.
- `Dense(k)` denotes a fully connected layer of width k.

Based on the above notations, we now describe the U-Net generator architecture with a sigmoid output function, and the corresponding critic. The training is performed using the Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.9$ and learning-rates 10^{-3}.The remaining hyper-parameters are listed in Table 1.

Architecture of cGAN for inferring conductivity
Generator architecture:

```
X →Conv(32,3) →LReLU →Res →X1 →Down(64) →Res(cin) →X2 →Down(128)
→Res(cin) →X3 →Down(256) →Res(cin) →X4 →Res(cin) →X5 →Up(256) →Res
→X6 →Up(128,X3) →Res(cin) →X7 →Up(64,X2) →Res(cin) →X8 →CC(X1)
→Conv(32,3) →LReLU →Res(cin) →Conv(1,3) →LReLU →Sigmoid →Output
```

Critic architecture:

```
[X;Y] →Down(32) →Res →Down(64) →Res(ln) →Down(128) →Res(ln)
→Down(256) →Res(ln) →Flatten →Dense(128) →LReLU →ln →Dense(1)
```

Table 1: WGAN hyper-parameters

Inferred field	Conductivity	Initial temperature
Training samples	8000	8000
Latent dimension	50	100
Epochs	1000	1000
Batch size	64	50
$n_{\text{critic}}/n_{\text{gen}}$	5	4

```
→Output
```

Architecture of cGAN for inferring initial temperature field
Generator architecture:

```
X →Conv(32,3) →LReLU →Res →X1 →Down(64) →Res(cin) →X2 →Down(128)
→Res(cin) →X3 →Res(cin) →X4 →Up(128) →Res →X5 →Up(64,X2) →Res(cin)
→X6 →CC(X1) →Conv(32,3) →LReLU →Res(cin) →Conv(1,3) →LReLU →Sigmoid
→Output
```

Critic architecture:

```
[X;Y] →Down(32) →Res →Down(64) →Res(ln) →Down(128) →Res(ln) →Flatten
→Dense(128) →LReLU →ln →Dense(1) →Output
```

B Heat conduction model and dataset generation

Consider the two-dimensional time-dependent heat conduction problem on a bounded domain $\Omega \subset \mathbb{R}^2$ with Dirichlet boundary conditions

$$\frac{\partial u(\boldsymbol{s},t)}{\partial t} - \nabla \cdot (\kappa(\boldsymbol{s}) \nabla u(\boldsymbol{s},t)) = b(\boldsymbol{s}), \qquad \forall\ (\boldsymbol{s},t) \in \Omega \times (0,T) \tag{5}$$

$$u(\boldsymbol{s},0) = m(\boldsymbol{s}), \qquad \forall\ \boldsymbol{s} \in \Omega \tag{6}$$

$$u(\boldsymbol{s},t) = 0, \qquad \forall\ (\boldsymbol{s},t) \in \partial\Omega \times (0,T). \tag{7}$$

Here u denotes the temperature field, κ denotes the spatially-varying conductivity field, b denotes the heat source and m denotes the initial temperature field. The steady-state problem corresponds to dropping the temporal dependence and the initial condition (6).

B.1 Datasets for inferring κ

We solve the non-linear coefficient inversion problem for the steady-state heat conduction problem posed on $\Omega = (0,1)^2$ for a constant heat source $b \equiv 10$. We assume that the measured temperature field is corrupted by noise, i.e., $\widetilde{u} = u + \eta$, where η is modelled as an uncorrelated Gaussian noise with zero mean and a covariance matrix given by $\sigma^2\mathbb{I}$. Here σ is taken to be 2.5% of the maximum value of u on the entire dataset.

To generate the training data for the GAN, the forward steady-state problem is solved using the Bubnov-Galerkin method with triangular first-order Lagrangian shape functions. The conductivity field is constructed as having a circular inclusion on a constant background of value unity. The position of the circle (s_1^c, s_2^c), the radius r and the conductivity value κ^c inside the circle were chosen from uniform distributions

$$s_1^c \sim \mathcal{U}(0,1), \quad s_2^c \sim \mathcal{U}(0,1), \quad r \sim \mathcal{U}(0.05, 0.3), \quad \kappa^c \sim \mathcal{U}(2,10).$$

The vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ of each training sample denote the values of the discretized κ and noisy temperature fields $\widetilde{u}$, respectively, interpolated on a 64×64 Cartesian grid.

We additionally construct two OOD test datasets; one with an ellipse and the other having two circles. The ellipses are parameterized by their center coordinates (s_1^e, s_2^e), aspect ratio r^e, angle of tilt θ, and conductivity κ^e

$$s_1^e \sim \mathcal{U}(0,1), \quad s_2^e \sim \mathcal{U}(0,1), \quad r^e \sim \mathcal{U}(0.3,1), \quad \theta \sim \mathcal{U}(0,\pi), \quad \kappa^e \sim \mathcal{U}(2,10).$$

For the double circles, we have their centers ((s_1^a, s_2^a) and (s_1^b, s_2^b)), radii (r^a and r^b), and conductivity values (κ^a and κ^b) sampled from

$$\begin{aligned} s_1^a &\sim \mathcal{U}(0,1), \quad s_2^a \sim \mathcal{U}(0,1), \quad r^a \sim \mathcal{U}(0.05, 0.25), \quad \kappa^a \sim \mathcal{U}(2,5), \\ s_1^b &\sim \mathcal{U}(0,1), \quad s_2^b \sim \mathcal{U}(0,1), \quad r^b \sim \mathcal{U}(0.05, 0.25), \quad \kappa^b \sim \mathcal{U}(2,5). \end{aligned}$$

B.2 Datasets for inferring the initial condition

We consider the time-dependent heat equation on the domain $\Omega = (0, 2\pi)^2$ with $b \equiv 0$ and $\kappa \equiv 0.2$. We assume that the measured temperature field at $t = T = 1$ is corrupted by an uncorrelated Gaussian noise with zero mean and identity covariance matrix. Given such a measurement, we wish to infer the initial temperature field m.

To construct training dataset, we consider the MNIST handwritten digits [18] and linearly scale the image intensity to lie between 0 and 4 units. This forms the discrete initial temperature field represented on a 28×28 Cartesian grid. The corresponding final temperature field on the same grid is obtained by solving (5)-(7) using a central-space-backward-time finite difference scheme. The vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ of each training sample denote the discretized initial and (noisy) final temperature fields, respectively.

We create an additional OOD test dataset, where the initial condition is represented using the notMNIST[1] dataset with the same linear scaling of the image intensity.

[1] Available at: http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html