We present vec2pix, a deep neural network designed to predict categorical or continuous 2D subsurface property fields from one-dimensional measurement data (e.g., time series), thereby, offering a new way to solve inverse problems. The method’s capabilities are investigated through two types of synthetic inverse problems: (a) a crosshole ground penetrating radar (GPR) tomography experiment where GPR travel times are used to infer a 2D velocity field, and (2) a multi-well pumping experiment within an unconfined aquifer for which time series of transient hydraulic heads serve to retrieve a 2D hydraulic conductivity field. For each problem type, both a multi-Gaussian and a binary channelized subsurface domain with long-range connectivity are considered. Using a training set of 20,000 examples, the method is found to always recover a 2D model that is in much closer agreement with the true model than the closest training model in the forward-simulated data space. Even if the recovered models by vec2pix are visually close to the true ones, the data misfits associated with their forward responses are generally larger than the noise level that was used to contaminate the true data. If fitting to the data noise level is required, then vec2pix-based inversion models can be used as initial inputs for more traditional multiple-point statistics inversion. Apart from further improving model reconstruction accuracy, more work is needed to provide an uncertainty assessment of the inferred models. Despite these current limitations, this study opens up a promising research avenue on how to use deep learning to infer subsurface models from indirect measurement data.
use off-the-shelf DL algorithms to train a network to turn geoscientific data vectors such as time-series into 2D or 3D subsurface models. In this contribution, we take the first steps towards a DL framework that takes one or multiple time series or other data represented in a data vector and map them into a subsurface model. Our presented examples focus on inferring permeability for given 2-D channelized and multi-Gaussian prior model, but the applicability of the approach is much wider than this. Our method builds on the influential image-to-image translation framework \cite{10} that we modified to ensure accurate data-to-model mapping. The results produced by our so-called vec2pix algorithm demonstrate the feasibility of 1D-to-2D transfer, which allows for many possible applications in hydrology, geophysics and Earth system science. Challenges remain, particularly those related to non-uniqueness, non-linearity and uncertainty quantification. Nevertheless, this work opens up the exciting prospect of deep-learning inverse mapping that turn environmental time-series or other data vectors into subsurface or surface models.

The remainder of this paper is organized as follows. Section 2 summarizes related work and how it differs from our method, before section 3 describes our proposed domain transfer network and its training, together with the considered inverse problems. This is followed by section 4 which presents our domain transfer inversion results. In section 5 we discuss our main findings and outline current limitations and possible future developments. Finally, section 6 provides a short conclusion.

2 Related Work

Inversion using an image-to-image domain transfer network has been proposed a few times in the context of 2D seismic inversion \cite{121} \cite{30} and recently for 2D travel time tomography by \cite{5}. We also consider 2D travel time tomography in some of our applications. Yet as opposed to \cite{5} who work within a 2D-to-2D transfer paradigm, we cast the problem within a vector-to-image transform framework. Other main differences between our work and the study by \cite{5} are that (i) we use a totally different neural network architecture, (ii) we consider higher dimensional and more complex geologic within a vector-to-image transform framework. Other main differences between our work and the study by \cite{5} are that (i) we use a totally different neural network architecture, (ii) we consider higher dimensional and more complex geologic prior models, and (iii) we use much less training examples for learning the weights and biases of our network: 20,000 training examples are used herein against as many as 2.5 millions for \cite{5}. In subsurface hydrology, the study by \cite{26} is a first step towards inverting steady-state groundwater flow data with image-to-image domain transfer networks. Both \cite{21} and \cite{26} added loss functions to the cycleGAN by \cite{34} to promote reconstruction of paired images. The network proposed herein also adds a reconstruction loss to a GAN algorithm but this time to honor the data vector to 2D model mapping. Doing so required substantial network design modifications. Furthermore, we consider both a cycleGAN framework, with joint learning of the data-to-model and model-to-data mappings (see section 3.1), and a simpler and computationally cheaper alternative where only the data-to-model transform is considered without model-to-data mapping and imposing cycle consistency (see section 3.2).

3 Methods

3.1 Vector-to-image transfer network within a CycleGAN framework

Let us denote by $Y$ the measurement data (vector) domain, and by $X$ the model (2D subsurface property field) domain. Similarly to cycleGAN \cite{34}, our full model consists of two mapping functions, $G_{YX}$ and $G_{XY}$, with associated discriminator functions, $D_X$ and $D_Y$, respectively:

$$
G_{YX} : \mathbb{R}^Y \rightarrow \mathbb{R}^X, D_X : \mathbb{R}^X \rightarrow [0, 1],
$$

$$
G_{XY} : \mathbb{R}^X \rightarrow \mathbb{R}^Y, D_Y : \mathbb{R}^Y \rightarrow [0, 1].
$$

The key operator here is $G_{YX}$, that predicts the model $\hat{x}$ corresponding to the measurement data vector $y$ it is fed with, $\hat{x} = G_{YX}(y)$, while $G_{XY}$ transfers a model $x$ into a data vector prediction, $\hat{y} = G_{XY}(x)$. The discriminator $D_X$ tries to distinguish between the true and predicted models, $x$ and $\hat{x}$, and $D_Y$ aims to distinguish between the true and predicted measurement vectors, $y$ and $\hat{y}$. At training time, $G_{YX}$, $G_{XY}$, $D_X$ and $D_Y$ are jointly learned using the sum of three losses: an adversarial loss, a cycle consistency loss and a reconstruction loss.

Following the original GAN proposed by \cite{6}, the adversarial loss for the $G_{YX}$ transfer function and its discriminator $D_X$ is given by

$$
\mathcal{L}_{GAN} (G_{YX}, D_X, y, x) = \mathbb{E}_{x \sim p_x} [\log (D_X(x))] + \mathbb{E}_{y \sim p_y} [\log (1 - D_X(G_{YX}(y)))] .
$$

(2)

The goal of the $G_{YX}$ function is to maximize equation (2) whereas the $D_X$ function tries to minimize it

$$
\min_{G_{YX}} \max_{D_X} \{ \mathcal{L}_{GAN} (G_{YX}, D_X, y, x) \} .
$$

(3)

A similar objective function is used for the pair \{ $G_{XY}$, $D_Y$ \}, resulting in

$$
\min_{G_{XY}} \max_{D_Y} \{ \mathcal{L}_{GAN} (G_{XY}, D_Y, x, y) \} .
$$

(4)
The cycle consistency loss introduced by [34] aims at ensuring that for each data vector \( \mathbf{y} \), the forward translation cycle, \( \tilde{\mathbf{y}} = G_{XY} (G_{YX} (\mathbf{y})) \), brings \( \mathbf{y} \) back to something that is close to the original \( \mathbf{y} \) vector. Similarly, \( G_{YX} \) and \( G_{XY} \) should satisfy backward translation cycle consistency, \( \tilde{\mathbf{x}} = G_{YX} (G_{XY} (\mathbf{x})) \approx \mathbf{x} \). The full cycle consistency loss takes the form [34]

\[
L_{\text{cyc}} (G_{YX}, G_{XY}) = E_{\mathbf{x} \sim p_{\mathbf{x}}} [||\mathbf{y} - G_{XY} (G_{YX} (\mathbf{y}))||_1] + E_{\mathbf{x} \sim p_{\mathbf{x}}} [||\mathbf{x} - G_{YX} (G_{XY} (\mathbf{x}))||_1].
\]  (4)

The losses described so far are equivalent those used by [34]. However, since we work with paired data \([\mathbf{x}_i, \mathbf{y}_i]\) and thus want to enforce accurate reconstruction in both the data-to-model mapping, \( \mathbf{x}_i = G_{YX} (\mathbf{y}_i) \approx \mathbf{x}_i \), and model-to-data mapping, \( \tilde{\mathbf{y}}_i = G_{XY} (\mathbf{x}_i) \approx \mathbf{y}_i \). Similarly as to the 2D-to-2D image transfer works by [10, 21, 26], we use the following reconstruction losses

\[
L_{\text{rec}} (G_{YX}, \mathbf{y}, \mathbf{x}_i) = E_{\mathbf{x} \sim p_{\mathbf{x}}} [||\mathbf{y} - G_{YX} (\mathbf{y})||_1]
\]

\[
L_{\text{rec}} (G_{XY}, \mathbf{y}, \mathbf{x}_i) = E_{\mathbf{x} \sim p_{\mathbf{x}}} [||\mathbf{y} - G_{XY} (\mathbf{x})||_1]
\]  (5)

Combining equations (2), (4) and (5), our full objective function used in vec2pix becomes

\[
L (G_{YX}, G_{XY}, D_X, D_Y) \geq L_{\text{GAN}} (G_{YX}, D_X, \mathbf{y}, \mathbf{x}) + L_{\text{GAN}} (G_{XY}, D_Y, \mathbf{x}, \mathbf{y})
\]

\[
+ \lambda_{\text{cyc}} L_{\text{cyc}} (G_{YX}, G_{XY}) + \lambda_{\text{rec}} L_{\text{rec}} (G_{YX}, \mathbf{y}, \mathbf{x}_i)
\]

\[
+ \lambda_{\text{rec}} L_{\text{rec}} (G_{XY}, \mathbf{y}, \mathbf{x}_i),
\]  (6)

where \( \lambda_{\text{cyc}} \), \( \lambda_{\text{rec}} \) and \( \lambda_{\text{cyc}} \) define the relative importance of each objective. Following [34], we set \( \lambda_{\text{cyc}} = 10 \). After preliminary testing with our reconstruction losses with the main goal of obtaining a \( G_{YX} \) transform that is as accurate as possible, we set \( \lambda_{\text{rec}} = 10^5 \) and \( \lambda_{\text{cyc}} = 10^4 \). Overall, our training procedure aims to solve

\[
\min_{G_{YX}, G_{XY}} \max_{D_X, D_Y} \{ L (G_{YX}, G_{XY}, D_X, D_Y) \}.
\]  (7)

3.2 Vector-to-image transfer network within a simple GAN framework

Note that in this work, we are not interested in the more classical model-to-data mapping provided by \( G_{XY} \). Ignoring the latter, the learning problem can be simplified as

\[
G_{YX} : \mathbb{R}^Y \rightarrow \mathbb{R}^X, D_X : \mathbb{R}^X \rightarrow [0, 1].
\]  (8)

and

\[
L (G_{YX}, D_X) = L_{\text{GAN}} (G_{YX}, D_X, \mathbf{y}, \mathbf{x}) + \lambda L_{\text{rec}} (G_{YX}, \mathbf{y}, \mathbf{x}_i),
\]  (9)

where \( \lambda \) defines the relative importance of the two objectives. After preliminary testing, we set \( \lambda = 10^4 \). The goal of training then reduces to

\[
\min_{G_{YX}} \max_{D_X} \{ L (G_{YX}, D_X) \}.
\]  (10)

We refer to this simpler one-way approach as one-way GAN.

3.3 Implementation

As stated earlier, the main methodological difference between our network architecture and those used previously within the 2D-to-2D transform paradigm [10, 34, 21, 26] is that our code processes a 1D (input) vector to output a 2D array. Our generator network architectures are based on those found in [34] which follow the recent state-of-the-art in computer vision. To make our vec2pix generator suitable to the 1D-to-2D \( (G_{YX}) \) domain transfer, we start by projecting the input data vector into an increasingly larger number of lower-dimensional representations (or latent spaces) using a series of 1D convolutions with increasingly larger number of channels (or filters, see Figure 1 and Appendix A). Then we apply a reshape operation to convert the final 1D representations into 2D representations before (i) further processing this information through a series of so-called “ResNet” residual blocks [9] and (ii) projecting the derived latent spaces into increasingly larger dimensional representations while reducing their numbers, until the final 2D model is produced. This using a combination of 2D transposed convolutions with a final 2D convolution (Figure 1). The key step of going from a 1D to a 2D domain therefore consists in the simple yet practical reshaping operation. Our discriminators are fully convolutional such as in [14]. Our generators and discriminators are detailed in Appendix A.

The Adam optimization solver [12] was used for training. Following [34], we used a learning rate of 0.0002 and values of 0.5 and 0.999 for the \( \beta_1 \) and \( \beta_2 \) momentum parameters. Unless stated otherwise, the number of epochs used in training is 150, and the batch size is 25. For every experiment, the training set comprises 20,000 examples of \( \mathbf{x}_i - \mathbf{y}_i \).
pairs, and an additional small validation set of 100 pairs (unseen by the training algorithm) is used to monitor the evolution of the different loss functions during training. The value of this validation loss was used to guide the selection of the trained model, particularly to avoid over-fitting. The indices of the input-output training pairs are shuffled at the beginning of every epoch to help the gradient descent escaping local minima. Furthermore, to make training robust to the noise in the data, that is, to account for the data measurement error during training, each true data vector used for training was corrupted with a new Gaussian white noise realization prior to the next epoch. With respect to performance evaluation, an independent test set made of 1000 examples was used to assess the performance of the proposed approach. Hence, inversion performance is assessed by evaluating how good each of those 1000 test models are when the trained $G_{YX}$ transformer is fed with the corresponding noise-contaminated data.

3.4 Synthetic Inverse Problems

To test vec2pix we consider inversion of both crosshole ground penetrating radar (GPR) data and transient pressure data during pumping. As of prior geologic models, we consider two common cases: a 2D multi-Gaussian prior and a 2D binary channelized aquifer prior. Regarding the latter, the DeeSse (DS) MPS algorithm [18] was used to generate the training and test models from the channelized aquifer TI proposed by [31]. To produce the multi-Gaussian realizations for training and test purposes, we used the circulant embedding method [4].

3.5 Crosshole GPR data

Our first type of data is travel times obtained from crosshole GPR tomography. Crosshole GPR imaging uses a transmitter antenna to emit a high-frequency electromagnetic wave at a location in one borehole and a receiver antenna to record the arriving energy at a location in another borehole. The considered measurement data are first-arrival traveltimes for several transmitter and receiver locations. These data contain information about the GPR velocity distribution between the boreholes. The GPR velocity primarily depends on dielectric permittivity, which is strongly influenced by volumetric water content and, consequently, porosity in saturated media. The considered model domain is of size $128 \times 64$ with a cell size of 0.1 m, and our setup consists of two vertical boreholes that are located 6.4 m apart placed at the left and right hand sides of the domain. Sources (left) and receivers (right) are located between 0.5 and 4 m.
12.5 m depth with 0.5 m spacing (Figures 2a and 2c), leading to a total dataset of $y = \mathbf{d}_{\text{GPR}}$ of 625 traveltimes. The forward nonlinear ray-based solver is simulated by means of the pyGIMLi toolbox [24] using the Dijkstra method and the measurement error used to corrupt the data is 0.5 ns.

For the binary channelized aquifer case, the channel and background facies are assigned a velocity of 0.06 m ns$^{-1}$ and 0.08 m ns$^{-1}$, respectively (Figure 2a). For the multi-Gaussian case, a zero-mean anisotropic Gaussian covariance model with a variance (sill) of 0.5, integral scales in the horizontal and vertical directions of 2 m (20 pixels) and 4 m (40 pixels), respectively, and anisotropy angle of $60^\circ$ was selected. The model realizations, were then scaled in $[-1, 1]$ using the minimum and maximum pixel values over the 20,000 training models before the following relationship was used to convert a scaled model, $\mathbf{x}$, into a velocity model, $\mathbf{x}_{VEL} = 0.06 + 0.02 (1 - \mathbf{x})$ m/ns (Figure 2a). For illustrative purposes, the simulated data vectors corresponding to the models depicted in Figures 2a and 2c are shown in Figures 3a and 3c.

### 3.6 Transient pumping data

Our second type of data consists of transient piezometric heads induced by pumping. The total number of pixels is $80 \times 128 = 10,240$, and the $80 \times 128$ aquifer case lies in the $x - y$ plane with a grid cell size of 1 m and a thickness of 10 m. For the binary channelized aquifer case, channel and matrix materials (see Figure 2d) are assigned hydraulic conductivity values, $K_x$, of $1 \times 10^{-3}$ m/s and $1 \times 10^{-5}$ m/s, respectively. For the multi-Gaussian case, the same geostatistical parameters as for the GPR setup are used for $\log_{10}(K_x)$, except that the mean is now -3 and the variance 0.1. The assumed specific storage and specific yield of the aquifer are 0.0003 m$^{-1}$ and 0.3 (-), respectively. The MODFLOW-NWT [22] code is used to simulate unconfined transient groundwater flow with no-flow boundaries at the upper and lower sides and a lateral head gradient of 0.01 (-) with water flowing in the $x$-direction. Four wells are sequentially extracting water for 20 days at a rate of 0.001 m$^3$/s (red dots in Figures 2b and d). The measurement data were formed by acquiring daily simulated heads in the 4 four pumping wells (red dots in Figures 2b and d) and 9 measuring wells (white crosses in Figures 2b and d) during the 80 days simulation period. The measurement data comprises $y = \mathbf{d}_{\text{Flow}}$ of 1040 heads. The measurement error used to contaminate these data with a Gaussian white noise is set to 0.01 m. To get a better idea of what the simulated data look like, we refer the reader to Figures 3b and 3d that display the concatenated data vectors corresponding to the models depicted in Figures 2b and 2d.

### 4 Results

As written above, we focus solely on vec2pix, the data-to-model transform provided by $G_{Y \rightarrow X}$. One can thus wonder why $G_{Y \rightarrow X}$ and $G_{X \rightarrow Y}$ are jointly learned within a cycleGAN framework. We have tested learning $G_{Y \rightarrow X}$ both within a cycleGAN-based scheme (see section 3.1) and using a simpler combination of a GAN loss and a reconstruction loss (“one-way GAN”, see section 3.2). Upon selection of the best training epoch, our cycleGAN-based learning of $G_{Y \rightarrow X}$ was found to provide slightly better results than those obtained when using the simpler learning approach, which is why we favor this approach despite its higher training cost. Importantly, all the results presented in this section are obtained from the cycleGAN-based learning of $G_{Y \rightarrow X}$.

For each case study we investigate the performance of $G_{X \rightarrow Y}$ based on the independent $i = 1, \cdots, 1000$ test pairs of model, $\mathbf{x}_i$, and data, $\mathbf{y}_i$. For the considered case studies, the reconstruction loss in validation (equation (4)) typically no longer improved during training after some 50 - 60 training epochs. For each test model, $\mathbf{x}_i$, the root-mean-square error (RMSE) between the associated data $\mathbf{y}_i$ and the $j = 1, \cdots, 20,000$ training data vectors $\mathbf{y}_j$ is computed and the minimum RMSE over the resulting 20,000 values is retained as the distance in data space between the considered test model and the training set. On this basis, we specifically compare the true and predicted model in cases where:

1. The true model is taken as the most different test model from the set of training models in the data space.
2. The true model is taken as the second most different test model from the set of training models in the data space.
3. The true model is randomly selected from the test set. This procedure is repeated three times.

Cases 1 and 2 serve to highlight the capacity of vec2pix to generalize for cases that are distinctively different from the training data. In total, this leads to five cases where differences between true models and those predicted by vec2pix are scrutinized. Besides, the complete distribution of 1000 RMSEs between the test data and the data simulated by feeding the forward solver with the models predicted by $G_{Y \rightarrow X}$ for these test data is also considered. In addition, two similarity indices between true and vec2pix models are computed for the 1000 test examples: the $l_1$ norm, $||\mathbf{x} - \mathbf{X}||_1$, and
Figure 2: Four considered synthetic case studies. Subfigures (a) and (c) depict the GPR tomography cases with (a) multi-Gaussian and (c) channelized bimodal surface structures. The red triangles and orange squares in subfigures (a) and (c) represent the GPR source and receiver positions, respectively. Subfigures (b) and (d) depict the transient pumping case with (b) multi-Gaussian and (d) channelized bimodal surface structures. The red dots and white crosses in subfigures (b) and (d) represent the pumping/observation and pure observation wells, respectively. The models displayed in subfigures (a-d) are randomly chosen from the 20,000 training models for each of the four examples.

and the structural similarity index (SSIM) \cite{28}

\[
SSIM(u, v) = \frac{2\mu_u\mu_v + c_12\sigma_{uv} + c_2}{\mu_u^2 + \mu_v^2 + c_1\sigma_u^2 + \sigma_v^2 + c_2},
\]

where \(u\) and \(v\) denote two \(N_p \times N_p\) windows subsampled from \(x\) and \(\tilde{x}\), respectively. \(\mu\) and \(\sigma^2\) are the mean and variance of \(u\) and \(v\), \(\sigma_{uv}\) represents the covariance between \(u\) and \(v\), and \(c_1 = 0.01\) and \(c_2 = 0.03\) are two small constants \cite{28}. Averaged over all \(u\) and \(v\) sliding windows, the mean SSIM ranges from -1 to 1, with 1 meaning that the two compared images are identical. Similarly as in the studies by \cite{25} and \cite{5}, we set \(N_p = 7\).
4.1 Case study 1: crosshole GPR data and multi-Gaussian domain

The vec2pix results for the GPR travel time tomography within a multi-Gaussian domain are presented in Figure 4 and Table 1 for the five selected true models. Table 2 lists the corresponding performance statistics for the 1000 test examples. It is observed that the produced vec2pix models always induce a lower data misfit and are more similar to
the true test models than the corresponding closest training models in data space (Tables 1 and 2). Indeed the vec2pix models display a two to three times smaller $l_1$-norm than the closest training models in data space (Table 1). In addition, the SSIMs of the vec2pix models are 15% to 25% larger than those of the closest training models in data space (Tables 1 and 2). Using 20,000 training examples, it is therefore a much better option to train and use vec2pix to invert the "measurement" data than to simply pick up the training model with the best corresponding fit to the data. This shows that vec2pix can generalize. The data RMSE of the forward simulated vec2pix realizations are globally in the 0.5 ns - 0.8 ns range with a median of 0.58 ns (Table 2), which is close to the "true" noise level of 0.5 ns used to contaminate the data ("true" measurement error). Overall, as compared to using the closest training model, vec2pix allows for a reduction in data RMSE of 2 to 3 times for the considered multi-Gaussian problem (Table 1). Unfortunately, a small but systematic artefact appears in the predicted models around $x = 3$ m and $z = 9.6$ m (Figure 4). This feature is discussed in section 5.

Note that this artefact could be removed by post-processing, but we leave it there to provide a fair assessment of the method.

Table 1: Analysis of the five selected true models for the case with crosshole GPR data and a multi-Gaussian domain. The following names are associated with subfigure names in Figure 4: True, Closest TR, Predicted #1 and Predicted #2. The (a - t) letters refer to the corresponding subfigures in Figure 4. RMSE$_{data}$ denotes the RMSE in data space described in the main text. The (a - d) letters represent the case when the true model is taken as the most different test model from the set of training models in the data space, the (e - h) letters are for the case when the true model is taken as the second most different test model from the set of training models in the data space, and the (i - l), (m - p), and (q - t) letters point to three cases when the true model is randomly selected from the test set. $l_1$ and SSIM are the $l_1$-norm and structural similarity index defined in the main text. $l_1$ is calculated in the velocity (m/ns) domain while SSIM is computed in the rescaled $[0, 1]$ domain.

| True model | RMSE$_{data}$ (ns) | $l_1$ (m/ns) | SSIM (-) |
|------------|---------------------|--------------|----------|
| (a) True   | 0.5                 | 0            | 1        |
| (b) Closest TR | 1.73              | 18.92        | 0.72     |
| (c) Predicted #1 | 0.66              | 7.96         | 0.92     |
| (d) Predicted #2 | 0.71              | 8.06         | 0.92     |
| (e) True   | 0.5                 | 0            | 1        |
| (f) Closest TR | 1.44              | 15.35        | 0.77     |
| (g) Predicted #1 | 0.58              | 5.91         | 0.92     |
| (h) Predicted #2 | 0.64              | 5.73         | 0.93     |
| (i) True   | 0.5                 | 0            | 1        |
| (j) Closest TR | 0.98              | 13.31        | 0.79     |
| (k) Predicted #1 | 0.60              | 5.21         | 0.94     |
| (l) Predicted #2 | 0.60              | 4.97         | 0.95     |
| (m) True   | 0.5                 | 0            | 1        |
| (n) Closest TR | 1.01              | 12.37        | 0.82     |
| (o) Predicted #1 | 0.56              | 7.89         | 0.92     |
| (p) Predicted #2 | 0.56              | 7.27         | 0.93     |
| (q) True   | 0.5                 | 0            | 1        |
| (r) Closest TR | 1.16              | 13.17        | 0.78     |
| (s) Predicted #1 | 0.55              | 6.03         | 0.93     |
| (t) Predicted #2 | 0.57              | 6.12         | 0.93     |

Table 2: Statistics of the distribution of the 1000 predicted models (Predicted) and closest training model in data space (Closest TR) associated with the 1000 examples of the test set for the case with crosshole GPR data and a multi-Gaussian domain. RMSE$_{data}$ is the RMSE in data space (see main text), $l_1$ is the $l_1$-norm and SSIM is the structural similarity index.

| Model       | Min | P10 | P25 | Median | P75 | P90 | Max |
|-------------|-----|-----|-----|--------|-----|-----|-----|
| Closest TR  | 0.77| 0.92| 0.97| 1.03   | 1.10| 1.16| 1.73|
| Predicted   | 0.51| 0.54| 0.55| 0.58   | 0.61| 0.65| 0.80|
| Closest TR  | 6.92| 10.57| 11.41| 12.36| 13.56| 14.56| 19.27|
| Predicted   | 3.84| 5.03| 5.59| 6.35   | 7.22| 8.14| 12.63|
| Closest TR  | 0.66| 0.75| 0.78| 0.80   | 0.83| 0.84| 0.91|
| Predicted   | 0.86| 0.91| 0.92| 0.93   | 0.94| 0.95| 0.96|
Figure 4: five selected true models (first column: True), associated closest training model in data space (second column: Closest TR), and predicted models (third and fourth columns: predicted #1 and predicted #2) for the case with crosshole GPR data and a multi-Gaussian domain. Each row in the figure corresponds to a different true model. The first row corresponds to the case where the true model is taken as the most different test model from the 20,000 training models in data space (see main text). The second row corresponds to the case where the true model is taken as the second most different test model from the 20,000 training models in data space. The third, fourth and fifth rows are for cases where the true model is randomly selected from the 1000 test models. The predicted #1 and predicted #2 models are obtained by presenting vec2pix with the same true “measurement” data contaminated with two different noise realizations. Table 1 lists the prediction quality statistics associated with the models displayed in the (a - t) subfigures.

4.2 Case study 2: crosshole GPR data and binary channelized domain

Results for the travel time tomography within a binary channelized domain are displayed in Figure 5, Table 3 and Table 4. These results are in line with those obtained for the multi-Gaussian case: the predicted test models show lower data RMSE, lower $l_1$ and larger SSIM statistics than the closest training models in data space. Also, the predicted test models look visually close to their true counterparts. With a median data RMSE of 0.82 ns and min and max data RMSEs of 0.58 ns and 1.74 ns (Table 4), the predicted test models induce data RMSE values that are significantly larger than the “true” noise level of 0.5 ns. Nevertheless, vec2pix permits reduction in data RMSE of a factor 2 to 3 compared to
selecting the closest training model (Table 3). The associated SSIM indices are also smaller than for the multi-Gaussian case: the P10 and median SSIM values are now 0.73 and 0.82 (Table 4) against 0.91 and 0.93 for the multi-Gaussian case (Table 2). The small deterministic artefact present in the continuous multi-Gaussian model predictions (around \( x = 3 \) m and \( z = 9.6 \) m, see Figure 4) is no longer found in the binary case. Overall, despite leading to larger data RMSEs compared to the prescribed noise level of 0.5 ns, the vec2pix models are fairly similar to the true ones (Figure 5 and Tables 3).

Figure 5: five selected true models (first column: True), associated closest training model in data space (second column: Closest TR), and predicted models (third and fourth columns: predicted #1 and predicted #2) for the case with crosshole GPR data and a binary channelized domain. Each row in the figure corresponds to a different true model. The first row corresponds to the case where the true model is taken as the most different test model from the 20,000 training models in data space (see main text). The second row corresponds to the case where the true model is taken as the second most different test model from the 20,000 training models in data space. The third, fourth and fifth rows are for cases where the true model is randomly selected from the 1000 test models. The predicted #1 and predicted #2 models are obtained by presenting vec2pix with the same true "measurement" data contaminated with two different noise realizations. Table 3 lists the prediction quality statistics associated with the models displayed in the (a - t) subfigures.
Table 3: Analysis of the five selected true models for the case with crosshole GPR data and a binary channelized domain. The following names are associated with subfigure names in Figure 5: True, Closest TR, Predicted #1 and Predicted #2. The (a - t) letters refer to the corresponding subfigures in Figure 5. RMSE_{data} denotes the RMSE in data space described in the main text. The (a - d) letters represent the case when the true model is taken as the most different test model from the set of training models in the data space, the (e - h) letters are for the case when the true model is taken as the second most different test model from the set of training models in the data space, and the (i - l), (m - p), and (q - t) letters point to three cases when the true model is randomly selected from the test set. $l_1$ and SSIM are the $l_1$-norm and structural similarity index defined in the main text. $l_1$ is calculated in the velocity (m/ns) domain while SSIM is computed in the rescaled [0, 1] domain.

| Model           | RMSE_{data} (ns) | $l_1$ (m/ns) | SSIM (-) |
|-----------------|------------------|--------------|----------|
| True            | 0.5              | 0            | 1        |
| (b) Closest TR  | 3.50             | 45.36        | 0.40     |
| (c) Predicted #1| 1.14             | 10.04        | 0.74     |
| (d) Predicted #2| 0.98             | 9.38         | 0.75     |
| (e) True        | 0.5              | 0            | 1        |
| (f) Closest TR  | 3.06             | 42.42        | 0.41     |
| (g) Predicted #1| 1.25             | 16.08        | 0.64     |
| (h) Predicted #2| 1.19             | 17.02        | 0.63     |
| (i) True        | 0.5              | 0            | 1        |
| (j) Closest TR  | 2.46             | 49.28        | 0.39     |
| (k) Predicted #1| 0.84             | 10.52        | 0.75     |
| (l) Predicted #2| 0.92             | 8.14         | 0.78     |
| (m) True        | 0.5              | 0            | 1        |
| (n) Closest TR  | 2.14             | 20.06        | 0.60     |
| (o) Predicted #1| 0.99             | 10.42        | 0.74     |
| (p) Predicted #2| 0.95             | 9.82         | 0.75     |
| (q) True        | 0.5              | 0            | 1        |
| (r) Closest TR  | 1.20             | 14.92        | 0.70     |
| (s) Predicted #1| 0.61             | 3.88         | 0.89     |
| (t) Predicted #2| 0.58             | 3.88         | 0.89     |

Table 4: Statistics of the distribution of the 1000 predicted models (Predicted) and closest training model in data space (Closest TR) associated with the 1000 examples of the test set for the case with crosshole GPR data and a multi-Gaussian domain. RMSE_{data} is the RMSE in data space (see main text), $l_1$ is the $l_1$-norm and SSIM is the structural similarity index.

| Model           | Min  | P10  | P25  | Median | P75  | P90  | Max  |
|-----------------|------|------|------|--------|------|------|------|
| RMSE_{data} (ns)|      |      |      |        |      |      |      |
| Closest TR      | 0.77 | 1.31 | 1.53 | 1.78   | 2.09 | 2.35 | 3.50 |
| Predicted       | 0.58 | 0.66 | 0.73 | 0.82   | 0.94 | 1.07 | 1.74 |
| RMSE_{l1} (m/ns)|      |      |      |        |      |      |      |
| Closest TR      | 4.82 | 13.88| 17.59| 22.42  | 28.97| 35.29| 56.74|
| Predicted       | 1.90 | 4.32 | 5.76 | 7.53   | 9.54 | 12.14| 25.18|
| SSIM (-)        |      |      |      |        |      |      |      |
| Closest TR      | 0.32 | 0.50 | 0.56 | 0.63   | 0.70 | 0.75 | 0.90 |
| Predicted       | 0.58 | 0.73 | 0.78 | 0.82   | 0.85 | 0.88 | 0.94 |

4.3 Case study 3: transient pressure data and multi-Gaussian domain

For the transient pumping experiment within a multi-Gaussian domain, the vec2pix models are again visually close to the true ones (Figure 6), even if a small artefact systematically occurs around $x = 8$ m and $y = 4$ m in the vec2pix models (Figure 6). The RMSEs in data space produced by the vec2pix models are overall similar to those produced by the closest training models (Tables 5 and 6), and are mostly distributed in the 0.02 m - 0.03 m range. This means that they are two to three times larger than the “true” noise level of 0.01 m. However, the model reconstruction statistics, $l_1$-norm and SSIM, are substantially better for the vec2pix models than for the closest training models in data space (Tables 5 and 6). Hence, the vec2pix models display 40% to 50% smaller $l_1$-norms and 10% to 25% larger SSIMs.
Table 5: Analysis of the five selected true models for the case with transient subsurface pressure data and a multi-Gaussian domain. The following names are associated with subfigure names in Figure 6: True, Closest TR, Predicted #1 and Predicted #2. The (a - t) letters refer to the corresponding subfigures in Figure 6. RMSE_{data} denotes the RMSE in data space described in the main text. The (a - d) letters represent the case when the true model is taken as the most different test model from the set of training models in the data space, the (e - h) letters are for the case when the true model is taken as the second most different test model from the set of training models in the data space, and the (i - l), (m - p), and (q - t) letters point to three cases when the true model is randomly selected from the test set. $l_1$ and SSIM are the $l_1$-norm and structural similarity index defined in the main text. $l_1$ is calculated in the log_{10} K_s (-) domain while SSIM is computed in the rescaled $[0, 1]$ domain.

| True model | RMSE_{data} (m) | $l_1$ (m) | SSIM (-) |
|------------|-----------------|-----------|-----------|
| (a) True   | 0.010           | 0         | 1         |
| (b) Closest TR | 0.060       | 2807      | 0.79      |
| (c) Predicted #1 | 0.054     | 1692      | 0.89      |
| (d) Predicted #2 | 0.062     | 1688      | 0.90      |
| (e) True   | 0.010           | 0         | 1         |
| (f) Closest TR | 0.049       | 2886      | 0.77      |
| (g) Predicted #1 | 0.071     | 1834      | 0.89      |
| (h) Predicted #2 | 0.067     | 1875      | 0.88      |
| (i) True   | 0.010           | 0         | 1         |
| (j) Closest TR | 0.027       | 2848      | 0.75      |
| (k) Predicted #1 | 0.021     | 1516      | 0.92      |
| (l) Predicted #2 | 0.023     | 1449      | 0.92      |
| (m) True   | 0.010           | 0         | 1         |
| (n) Closest TR | 0.022       | 3575      | 0.76      |
| (o) Predicted #1 | 0.016     | 1867      | 0.88      |
| (p) Predicted #2 | 0.022     | 1921      | 0.87      |
| (q) True   | 0.010           | 0         | 1         |
| (r) Closest TR | 0.023       | 3384      | 0.66      |
| (s) Predicted #1 | 0.020     | 1908      | 0.88      |
| (t) Predicted #2 | 0.019     | 2045      | 0.87      |

Table 6: Statistics of the distribution of the 1000 predicted models (Predicted) and closest training model in data space (Closest TR) associated with the 1000 examples of the test set for the case with transient subsurface pressure data and a multi-Gaussian domain. RMSE_{data} is the RMSE in data space (see main text), $l_1$ is the $l_1$-norm and SSIM is the structural similarity index.

| Model   | Min  | P10  | P25  | Median | P75  | P90  | Max  |
|---------|------|------|------|--------|------|------|------|
| Closest TR | 0.015| 0.019| 0.021| 0.023  | 0.026| 0.029| 0.061|
| Predicted | 0.012| 0.015| 0.017| 0.021  | 0.026| 0.033| 0.072|
| Closest TR | 1819 | 2408 | 2616 | 2848   | 3088 | 3321 | 3984 |
| Predicted | 1307 | 1514 | 1618 | 1747   | 1878 | 2012 | 2521 |
| Closest TR | 0.58 | 0.71 | 0.74 | 0.76   | 0.79 | 0.81 | 0.86 |
| Predicted | 0.81 | 0.86 | 0.87 | 0.89   | 0.90 | 0.91 | 0.93 |

4.4 Case study 4: transient pressure data and binary channelized domain

The hydraulic case with a binary channelized domain is by far the most challenging among the four considered ones. Indeed, the relationship between a binary channelized model and the resulting simulated transient flow data is highly nonlinear. As a consequence, across the 20,000 training models, the signal-to-noise-ratio (SNR) defined as the ratio of the average RMSE obtained by drawing prior realizations from the training image by MPS simulation to the noise level is in the 60 - 100 range. For this case, the vec2pix models are in better visual agreement with the true model than the closest training models in data space (Figure 7). This is confirmed by two to three times smaller $l_1$-norms and 10% to 70% larger SSIM indices (Tables 7 and 8). As for the GPR experiment within a binary channelized domain, no systematic artefact is observed in the vec2pix models. Even if vec2pix produces models that are of much better quality than the closest training models in data space, the resulting RMSEs in data space are often not better than those produced by the closest training models in data space. This is because a change of facies in the surroundings of a pumping well (red dots in Figure 2d) can dramatically affect the corresponding simulated data.
Table 7: Analysis of the five selected true models for the case with transient subsurface pressure data and a binary channelized domain. The following names are associated with subfigure names in Figure 7: True, Closest TR, Predicted #1 and Predicted #2. The (a - t) letters refer to the corresponding subfigures in Figure 7. RMSE_data denotes the RMSE in data space described in the main text. The (a - d) letters represent the case when the true model is taken as the most different test model from the set of training models in the data space, the (e - h) letters are for the case when the true model is taken as the second most different test model from the set of training models in the data space, and the (i - l), (m - p), and (q - t) letters point to three cases when the true model is randomly selected from the test set. \( l_1 \) and SSIM are the \( l_1 \)-norm and structural similarity index defined in the main text. \( l_1 \) is calculated in the \( \log_{10} K_s \) domain while SSIM is computed in the rescaled \([0, 1]\) domain.

| True model | RMSE_{data} (m) | \( l_1 \) (m) | SSIM (-) |
|------------|----------------|----------------|----------|
| (a) True   | 0.010          | 0              | 1        |
| (b) Closest TR | 0.385 | 9908           | 0.21     |
| (c) Predicted #1 | 0.229 | 2634           | 0.66     |
| (d) Predicted #2 | 0.370 | 2414           | 0.67     |
| (e) True   | 0.010          | 0              | 1        |
| (f) Closest TR | 0.281 | 8180           | 0.28     |
| (g) Predicted #1 | 0.132 | 2758           | 0.61     |
| (h) Predicted #2 | 0.215 | 2102           | 0.65     |
| (i) True   | 0.010          | 0              | 1        |
| (j) Closest TR | 0.052 | 8028           | 0.32     |
| (k) Predicted #1 | 0.275 | 2580           | 0.66     |
| (l) Predicted #2 | 0.068 | 2312           | 0.69     |
| (m) True   | 0.010          | 0              | 1        |
| (n) Closest TR | 0.037 | 2754           | 0.71     |
| (o) Predicted #1 | 0.019 | 1418           | 0.81     |
| (p) Predicted #2 | 0.016 | 1400           | 0.80     |
| (q) True   | 0.010          | 0              | 1        |
| (r) Closest TR | 0.027 | 3398           | 0.68     |
| (s) Predicted #1 | 0.016 | 1512           | 0.82     |
| (t) Predicted #2 | 0.014 | 1542           | 0.82     |

Table 8: Statistics of the distribution of the 1000 predicted models (Predicted) and closest training model in data space (Closest TR) associated with the 1000 examples of the test set for the case with transient subsurface pressure data and a binary channelized domain. RMSE_{data} is the RMSE in data space (see main text), \( l_1 \) is the \( l_1 \)-norm and SSIM is the structural similarity index.

| Model      | Min   | P10  | P25  | Median | P75  | P90  | Max   |
|------------|-------|------|------|--------|------|------|-------|
|            | RMSE_{data} (m) | \( l_1 \) (m) | SSIM (-) |
| Closest TR | 0.011 | 0.024 | 0.030 | 0.043  | 0.064 | 0.092 | 0.385 |
| Predicted  | 0.011 | 0.016 | 0.020 | 0.033  | 0.088 | 0.204 | 0.534 |
| Closest TR | 548   | 2335 | 3251 | 4359   | 5664.50 | 7049 | 11160 |
| Predicted  | 242   | 1054 | 1477 | 1971   | 2635  | 3274 | 6318  |
| Closest TR | 0.11  | 0.37  | 0.47  | 0.55   | 0.64  | 0.72  | 0.89  |
| Predicted  | 0.38  | 0.61  | 0.67  | 0.73   | 0.78  | 0.84  | 0.94  |
Figure 6: five selected true models (first column: True), associated closest training model in data space (second column: Closest TR), and predicted models (third and fourth columns: predicted #1 and predicted #2) for the case with transient subsurface pressure data and a multi-Gaussian domain. Each row in the figure corresponds to a different true model. The first row corresponds to the case where the true model is taken as the most different test model from the 20,000 training models in data space (see main text). The second row corresponds to the case where the true model is taken as the second most different test model from the 20,000 training models in data space. The third, fourth and fifth rows are for cases where the true model is randomly selected from the 1000 test models. The predicted #1 and predicted #2 models are obtained by presenting vec2pix with the same true “measurement” data contaminated with two different noise realizations. Table 5 lists the prediction quality statistics associated with the models displayed in the (a - t) subfigures.
Figure 7: five selected true models (first column: True), associated closest training model in data space (second column: Closest TR), and predicted models (third and fourth columns: predicted #1 and predicted #2) for the case with transient subsurface pressure data and a binary channelized domain. Each row in the figure corresponds to a different true model. The first row corresponds to the case where the true model is taken as the most different test model from the 20,000 training models in data space (see main text). The second row corresponds to the case where the true model is taken as the second most different test model from the 20,000 training models in data space. The third, fourth and fifth rows are for cases where the true model is randomly selected from the 1000 test models. The predicted #1 and predicted #2 models are obtained by presenting vec2pix with the same true “measurement” data contaminated with two different noise realizations. Table 7 lists the prediction quality statistics associated with the models displayed in the (a - t) subfigures.
5 Discussion

We have introduced vec2pix, a deep neural network for predicting 2D subsurface property fields from one-dimensional measurement data (e.g., time series). Our approach is illustrated using (1) synthetic GPR travel times for recovering a 2D velocity field, and (2) time series of transient hydraulic heads to infer a 2D hydraulic conductivity field. For each problem, both a multi-Gaussian and a binary channelized subsurface domain with long-range connectivity are considered. Training vec2pix is here achieved using 20,000 training examples. For every considered case, our method is found to retrieve a model that is much closer to the true model than the closest training model in data space. Although these recovered models generally look similar to the true models, the data RMSE obtained when forward simulating the vec2pix models are higher than the prescribed noise level (that is, Gaussian white noise used to contaminate the true data). In such case the inversion cannot be declared entirely successful. This is particularly true for our fourth case study that considers a transient pumping experiment within a channelized subsurface domain for which the relationship between model and simulated pressure data is highly nonlinear and, to some extent, not unique. If data fitting to the noise level is needed, we suggest that the final solution could be used as a starting point for a multiple-point statistics based inversion such as sequential geostatistical resampling (e.g., [19]). In addition, the following remarks are in order.

Moderately extensive testing with our simpler one-way GAN approach described in section 3.2 (that is, combining a GAN loss with a reconstruction loss) showed globally similar data-to-model mapping results to what is obtained when using our cycleGAN-based learning (see section 3.1) presented above. Nevertheless, a closer look revealed that, upon selection of the best epoch by means of our validation loss, the cycleGAN-based learning appears to induce 2% to 5% larger SSIM values and a similar 2% to 5% reduction in l_1-norm for the recovered models compared to the one-way learning. The additional regularization of G_{Y,X} provided by the cycle consistency constraints, \( x \approx G_{Y,X} (G_{X,Y} (x)) \approx x \) and \( \hat{y} = G_{X,Y} (G_{Y,X} (y)) \approx y \) thus seems to improve learning to some extent. However, more exhaustive testing is required to formally establish the possible added value of using a cycleGAN framework to learn \( G_{Y,X} \), compared to simply adding a reconstruction loss to a GAN loss. Regarding computational expense, for our available GPU resources cycleGAN-based learning incurs a 3.5 times larger computational time per training epoch than the one-way procedure.

A small artefact appears in the vec2pix models for the multi-Gaussian cases. GANs and their variants are notoriously hard to train as training is known to be prone to issues such as instability (the GAN parameters do not converge but keep oscillating), mode collapse (the variability of the generated samples is insufficient), overfitting and diminished gradients (e.g., when the generator gradient vanishes as the discriminator becomes too good). We are not sure what exactly causes the observed artefact in our continuous model predictions, but it might be related to the zero-padding of the input data vectors (see Appendix A). Although there is some evidence that GAN training procedures might not perform very differently from each others [16], more advanced training than the classical vanilla GAN used herein, such as the Wasserstein GAN [2] and its follow-ups [8], might improve the situation.

Besides improving the deterministic prediction made by our approach, more work is also needed to account for the uncertainty caused by the non-uniqueness in the model to data relationship and measurement data error. A simple way to do that could be to produce an ensemble of model predictions by randomly perturbing the measurement vector by different data error realizations before running vec2pix, to assess how random observational noise translates into uncertainty in model parameters.

The SSIM (equation (11)) is fully differentiable and it could then be used as a reconstruction loss instead of the l_1-norm. For our considered case studies, this strategy led to very unstable training and a deceiving performance of the data-to-model mapping. Nevertheless, other options are available, such as combining a multi-scale variant of the SSIM, MS-SSIM, with the l_1-norm [33]. This warrants further investigations.

A training set of 20,000 examples was used this study. Considering a larger training base would likely improve prediction quality, but at the cost of a larger computational demand. Globally, most of the computational time required for solving hydrologic and geophysical inverse problems is caused by running the forward solver. If hundreds of thousands of forward solver evaluations can be performed, then more conventional inverse methods may become more attractive than training a deep neural network. This is why no more than 20,000 training samples were considered herein.

For a real application, the measurements presented to vec2pix will be contaminated with a measurement error. That is why we trained vec2pix with noise-contaminated data. However, limited testing showed that corrupting the training data or not does not seem to lead to important differences at test time, when the test data are noise-contaminated. That said, we have used realistic, but low, noise levels to corrupt our data and the situation will likely change if larger noise values are prescribed.

Model-to-data mapping, also known as surrogate or proxy modeling, can be achieved by learning \( G_{X,Y} \) using both approaches described in sections 3.1 and 3.2. This will be the scope of a future study.
6 Conclusion

We introduce vec2pix, a deep neural network for predicting categorical or continuous 2D subsurface property fields from one-dimensional measurement data (e.g., time series) and, thereby, offering an alternative approach to solve inverse problems. The method is illustrated using (1) synthetic GPR travel times to infer a 2D velocity field, and (2) time series of transient hydraulic heads to retrieve a 2D hydraulic conductivity field. For each problem, both a multi-Gaussian and a binary channelized subsurface domain with long-range connectivity are considered. Using a training set of 20,000 examples, our approach is shown to always recover a 2D model that is much closer to the true model than the closest training model in the forward-simulated data space. Even though the inferred models look generally similar to the true ones, the data missfits obtained when forward simulating these models are in most cases larger than the noise level that was used to corrupt the true data. This implies that the vec2pix-based inversion cannot be deemed completely successful for the considered case studies. Besides improving model reconstruction accuracy, further work is needed to provide an uncertainty assessment of the inferred models. Despite these current limitations, we believe that this work opens up a promising research avenue on how to use deep learning to infer subsurface models from indirect measurement data.

7 Code availability

Upon acceptance of the manuscript, the code of our approach will be made available at https://github.com/elaloy/vec2pix

8 Appendix A: Network details

The $G_{XY}$ and $G_{XY}$ networks are made of convolutions, transposed convolutions and a series of “ResNet” residual blocks [2]. We use 6 residual blocks for cases involving binary images (or models) and 9 residual blocks for cases involving continuous images. Our used activation functions are either rectified linear unit: ReLU ($\text{max}(0,x)$) or hyperbolic tangent: Tanh, and we use reflection padding in the first and last layers of $G_{XY}$ and $G_{XY}$. Let $c_{2d7} - s1 - k_{in1} - k_{out} 64 - p0$ denote a $7 \times 7$ 2D Convolution-InceptionNorm-ReLU layer with $k_{in} = 1$ incoming channels (or filters), $k_{out} = 64$ outgoing channels, stride 1 and zero padding. We call $c_{2d7} - s1 - k_{in1} - k_{out} 64 - p0$ the same layer without normalization and with a Tanh activation function. Furthermore, $t_{c2d}$ signifies a 2D Transposed Convolution-InceptionNorm-ReLU, $op1$ means output padding of 1 and $R_{c2d} - k_{512}$ represents a residual block that contains two $3 \times 3$ 2D convolutional layers with InstanceNorm and $k = 512$ channels on both layers, and a ReLU activation function on the first layer. Lastly, $Re(z_r, z_c)$ and Fla mean reshaping a vector into a $z_r \times z_c$ array and flattening an 2D array, respectively. From input to output layer, our generators are built as follows

- $G_{XY}$:
  \[
  \begin{align*}
  &c_{1d7} - s1 - k_{in1} - k_{out} 64 - p0], [c_{1d3} - s2 - k_{in} 64 - k_{out} 128 - p1], [c_{1d3} - s2 - k_{in} 128 - k_{out} 256 - p1], \\
  &c_{1d3} - s2 - k_{in} 256 - k_{out} 512 - p1], Re( z_r, z_c), N_{res} \times [R_{2d} - k_{512}], \\
  &t_{c2d3} - s2 - k_{in} 512 - k_{out} 256 - p1 - op1], [t_{c2d3} - s2 - k_{in} 256 - k_{out} 128 - p1 - op1], [t_{c2d3} - s2 - k_{in} 64 - k_{out} 64 - p1 - op1].
  \end{align*}
  \]

- $G_{XY}$:
  \[
  \begin{align*}
  &c_{2d7} - s1 - k_{in1} - k_{out} 64 - p0], [c_{2d3} - s2 - k_{in} 64 - k_{out} 128 - p1], [c_{2d3} - s2 - k_{in} 128 - k_{out} 256 - p1], \\
  &c_{2d3} - s2 - k_{in} 256 - k_{out} 512 - p1], Fla, N_{res} \times [R_{1d} - k_{512}], [t_{c1d3} - s2 - k_{in} 512 - k_{out} 256 - p1 - op1], \\
  &t_{c1d3} - s2 - k_{in} 256 - k_{out} 128 - p1 - op1], [t_{c1d3} - s2 - k_{in} 128 - k_{out} 64 - p1 - op1], \\
  &[t_{c1d3} - s2 - k_{in} 64 - k_{out} 64 - p1 - op1].
  \end{align*}
  \]

where $z_r = \frac{X_r}{X}$ and $z_c = \frac{X_c}{X}$ with $X_r$ and $X_c$ the numbers of rows and columns of a model $X$, the incoming data vector $y$ is padded with zeros such as its size matches $\frac{X_r X_c}{X}$, and $N_{res}$ is the selected number of residual blocks (6 or 9, see above).

For the discriminators we use fully convolutional networks similarly as in [11],[14]. Adopting the same terminology as above, we have

- $D_X$:
  \[
  \begin{align*}
  &c_{2d3} - s2 - k_{in1} - k_{out} 64 - p1], [c_{2d3} - s2 - k_{in} 64 - k_{out} 128 - p0], [c_{2d3} - s2 - k_{in} 128 - k_{out} 256 - p1], \\
  &c_{2d3} - s2 - k_{in} 256 - k_{out} 512 - p1].
  \end{align*}
  \]

- $D_Y$:
  \[
  \begin{align*}
  &c_{1d3} - s2 - k_{in1} - k_{out} 64 - p1], [c_{1d3} - s2 - k_{in} 64 - k_{out} 128 - p0], [c_{1d3} - s2 - k_{in} 128 - k_{out} 256 - p1], \\
  &c_{1d3} - s2 - k_{in} 256 - k_{out} 512 - p1].
  \end{align*}
  \]
where the activation functions are now leaky ReLU with slope 0.2, normalization is not used for the first and last layer of each network, and the last layer of each network has a sigmoid activation function for the “vanilla” GAN loss function in equation (2). If a different loss function is used, such as a Wasserstein loss \cite{2} or a least-squares loss \cite{17}, then the last layer of $D_X$ and $D_Y$ has a linear activation function. For our considered case studies, limited testing did not show any substantial difference in training performance between using equation (2) and a least-squares loss. For a GAN loss, $L_{GAN} (G_{XY}, D_Y, x, y)$, the least-squares loss consists of minimizing $\mathbb{E}_{x \sim p_x} [(D_Y (G_{XY} (x)) - 1)^2]$ when $G_{XY}$ is trained, and minimizing $\mathbb{E}_{y \sim p_y} [(D_Y (y) - 1)^2] + \mathbb{E}_{x \sim p_x} [(D_Y (G_{XY} (x)))^2]$ when $D_Y$ is trained.

References

\cite{1} Araya-Polo, M., J. Jennings, A. Adler, and T. Dahlke, 2018. Deep-learning tomography. The Leading Edge, 37(1):58–66. https://library.seg.org/doi/10.1190/tle37010058.1.

\cite{2} Arjovsky, M., Chintala, S., and L. Bottou, 2017. Wasserstein GAN. arXiv preprint \url{arXiv:1701.07875}.

\cite{3} Chan, S., and A. Elsheikh, 2019. Parametric generation of conditional geological realizations using generative neural networks. Computational Geosciences, 23(5), 925 –952. https://doi.org/10.1007/s10596-019-09850-7.

\cite{4} Dietrich, C. R., and G. H. Newsam, 1997. Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix. SIAM Journal on Scientific Computing, 18, 1088–1107.

\cite{5} Earp, S., and A. Curtis, 2019. Probabilistic neural-network based 2D travel time tomography. arXiv preprint \url{arXiv:1907.00541}.

\cite{6} Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Y. Bengio, 2014. Generative adversarial networks. The Annual Conference on Neural Information Processing Systems (NIPS), Montréal, 2014.

\cite{7} Goodfellow, I., Bengio, Y., Courville, A. Deep learning. MIT Press; 2016. \url{http://www.deeplearningbook.org}.

\cite{8} Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and A. Courville, 2017. Improved training of Wasserstein GANs. arXiv preprint \url{arXiv:1704.00028}.

\cite{9} He, K., Zhang, X., Ren, S., and J. Sun, 2016. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), http://dx.doi.org/10.1109/CVPR.2016.90. Also available as arXiv preprint \url{arXiv:1512.03385}.

\cite{10} Isola, P., Zhu, J.-Y., Zhou, T., and A. A. Efros, 2016. Image-to-image translation with conditional adversarial networks. arXiv preprint \url{arXiv:1611.07004}.

\cite{11} Jetc, N., Bergmann, U., and R. Vollgraf, 2016. Texture synthesis with spatial generative adversarial networks. arXiv preprint \url{arXiv:1611.08207}.

\cite{12} Kingma, D. P., and J. L. Ba, 2015. ADAM: a method for stochastic optimization, The International Conference on Learning Representations (ICLR), San Diego.

\cite{13} Laloy, E., R. Héraut, J. Lee, D. Jacques, and N. Linde, 2017. Inversion using a new low-dimensional representation of complex binary geological media based on a deep neural network. Advances in Water Resources, 110, 387–405. https://doi.org/10.1016/j.advwatres.2017.09.029.

\cite{14} Laloy, E., R. Héraut, R., D. Jacques, and N. Linde, 2018. Training-image based geostatistical inversion using a spatial generative adversarial neural network. Water Resour. Res. 54(1):381–406, 2018. https://doi.org/10.1002/2017WR022148.

\cite{15} Laloy, E., N. Linde, C. Ruffino, R. Héraut, G. Gasso and D. Jacques, 2019. Gradient-based deterministic inversion of geophysical data with Generative Adversarial Networks: is it feasible? Computers and Geosciences, 23(5), 1193–1215, https://doi.org/10.1016/s1095-9649(18)30743-2.

\cite{16} Lucic, M., Kurach, K., Michalski, M., Gelly, S., and O. Bousquet, 2017. Are GANs created aqual? A large-scale study. arXiv preprint \url{arXiv:1711.10337}.

\cite{17} Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and S. P. Smolley, 2017. Least squares generative adversarial networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), http://dx.doi.org/10.1109/ICCV.2017.304. Also available as arXiv preprint \url{arXiv:1611.04076}.

\cite{18} Mariethoz, G., Renard, P., and J. Straubhaar, 2010. The Direct Sampling method to perform multiple-point geostatistical simulations. Water Resources Research, 46, W11536. \url{http://dx.doi.org/10.1029/2008WR007621}.
[19] Mariethoz, G., P. Renard, and J. Caers, 2010. Bayesian inverse problem and optimization with iterative spatial resampling. Water Resources Research, 46, W11530. [http://dx.doi.org/10.1029/2010WR009274](http://dx.doi.org/10.1029/2010WR009274).

[20] Mosser, L., D. Dubrule, O., and M.J. Blunt, 2017. Reconstruction of three-dimensional porous media using generative adversarial neural networks. Physical Review E., 96. https://doi.org/10.1103/PhysRevE.96.043309.

[21] Mosser, L., W. Kimman, J. Dransch, S. Purves, A. De la Fuente, and G. Ganssle, 2018. Rapid seismic domain transfer: Seismic velocity inversion and modeling using deep generative neural networks, arXiv preprint [arXiv:1805.08826](https://arxiv.org/abs/1805.08826).

[22] Niswonger, R.G., Sorab, P., and I. Motomu, 2011. MODFLOW-NWT, A Newton formulation for MODFLOW-2005. U.S. Geological Survey Techniques and Methods 6-A37, 44 p.

[23] Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., and Prabhat, 2019. Deep learning and process understanding for data-driven Earth system science. Nature, 566(7743), 195–204. [https://doi.org/10.1038/s41586-019-0912-1](https://doi.org/10.1038/s41586-019-0912-1).

[24] Rücker, C., Günther, T., and F.M. Wagner, 2017. pyGIMLi: An open-source library for modelling and inversion in geophysics. Computers and Geosciences, 109, 106–123. https://doi.org/10.1016/j.cageo.2017.07.011.

[25] Shen, C., 2018. A transdisciplinary review of deep learning research and its relevance for water resources scientists. Water Resources Research, 54(11), 8558–8593. https://doi.org/10.1029/2018WR022643.

[26] Sun, A. Y., 2018. Discovering state-parameter mappings in subsurface models using generative adversarial networks. Geophysical Research Letters, 45: 11137–11146. [https://doi.org/10.1029/2018GL080404](https://doi.org/10.1029/2018GL080404).

[27] Sun, A. Y., and B. R. Scanlon, 2019. How can Big Data and machine learning benefit environment and water management: a survey of methods, applications, and future directions. Environmental Research Letters, 14, 073001. [https://doi.org/10.1088/1748-9326/ab1b7d](https://doi.org/10.1088/1748-9326/ab1b7d).

[28] Wang, Z., Bovik, A. C., Sheikh, H. R., and E. P. Simoncelli, 2004. Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing, 13(4), 600–612.

[29] Wang, Y. Teng, Q., He, X., Feng, J., and T. Zhang, 2019. CT-image of rock samples super resolution using 3D convolutional neural network. Computers & Geosciences, 133, 104314. [https://doi.org/10.1016/j.cageo.2019.104314](https://doi.org/10.1016/j.cageo.2019.104314).

[30] Yang, F. and J. Ma, 2019. Deep-learning inversion: A next-generation seismic velocity model building method, 2019. Geophysics, 84(4):1JA–Z21. [https://doi.org/10.1190/geo2018-0249.1](https://doi.org/10.1190/geo2018-0249.1).

[31] Zahner, T., T. Lochbühler, G. Mariethoz, and N. Linde, 2016. Image synthesis with graph cuts: a fast model proposal mechanism in probabilistic inversion, Geophysical Journal International, 204(2), 1179–1190. [http://dx.doi.org/10.1093/gji/ggv517](http://dx.doi.org/10.1093/gji/ggv517).

[32] Zhang, L., Zhang, L., and B. Du, 2016. Deep learning for remote sensing data: a technical tutorial on the state of the art. IEEE Geoscience and Remote Sensing Magazine, 4(2), 22–40. [http://dx.doi.org/10.1109/MGRS.2016.2540798](http://dx.doi.org/10.1109/MGRS.2016.2540798).

[33] Zhao, H., Gallo, O., Frosio, I., and J. Kautz, 2017. Loss Functions for Image Restoration With Neural Networks. IEEE Transactions on Computational Imaging, 3(1), 47–57. [http://dx.doi.org/10.1109/TCI.2016.2644865](http://dx.doi.org/10.1109/TCI.2016.2644865).

[34] Zhu, J., T. Park, P. Isola, and A. A. Efros, 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks: IEEE International Conference on Computer Vision (ICCV), 2242–2251. arXiv preprint [arXiv:1703.10593](https://arxiv.org/abs/1703.10593).

[35] Zhu, Y., N. Zabaras, 2018. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. Journal of Computational Physics, 366, 415–437. [https://doi.org/10.1016/j.jcp.2018.04.018](https://doi.org/10.1016/j.jcp.2018.04.018).