Probabilistic Neural-Network Based 2D Travel Time Tomography

Neural Computing and Applications
https://doi.org/10.1007/s00521-020-04921-8

ORIGINAL ARTICLE

Probabilistic neural network-based 2D travel-time tomography

Stephanie Earp[1] · Andrew Curtis[1,2]

Received: 27 June 2019 / Accepted: 6 April 2020

Abstract
Travel-time tomography for the velocity structure of a medium is a highly nonlinear and nonunique inverse problem. Monte Carlo methods are becoming increasingly common choices to provide probabilistic solutions to tomographic problems but those methods are computationally expensive. Neural networks can often be used to solve highly nonlinear problems at a much lower computational cost when multiple inversions are needed from similar data types. We present the first method to perform fully nonlinear, rapid and probabilistic Bayesian inversion of travel-time data for 2D velocity maps using a mixture density network. We compare multiple methods to estimate probability density functions that represent the tomographic solution, using different sets of prior information and different training methodologies. We demonstrate the importance of prior information in such high-dimensional inverse problems due to the curse of dimensionality: unrealistically informative prior probability distributions may result in better estimates of the mean velocity structure; however, the uncertainties represented in the posterior probability density functions then contain less information than is obtained when using a less informative prior. This is illustrated by the emergence of uncertainty loops in posterior standard deviation maps when inverting travel-time data using a less informative prior, which are not observed when using networks trained on prior information that includes (unrealistic) a priori smoothness constraints in the velocity models. We show that after an expensive program of network training, repeated high-dimensional, probabilistic tomography is possible on timescales of the order of a second on a standard desktop computer.

Keywords Neural networks · Mixture density networks · Uncertainty estimation · Seismic tomography

1 Introduction

Seismic travel-time tomography is often used to reconstruct images of the interior of the Earth [1–3], but is a significantly nonlinear and nonunique inverse problem. To find solutions with minimal computation, the physics relating local wave speed to measured travel times is usually simplified by linearisation [4], but this creates large differences between linearised and true probabilistic solutions [5, 9]. Increases in compute power now allow fully nonlinear Monte Carlo sampling solutions to be found without linearisation, to solve problems in 2D [5, 6] and 3D [7–10]. Using Bayesian methods, such solutions provide samples (example tomographic models) that fit the data to within their measurement uncertainties, are consistent with available prior information and are distributed according to the posterior probability density function (pdf) across the parameter space; this pdf constitutes the full solution of tomographic problems. Nevertheless, such solutions are acquired at significant expense, typically requiring weeks of compute time for realistic data sets and expensive storage of large sample sets.

An alternative approach to estimate the posterior pdf is to use prior sampling [11, 12]. In this case, samples are created before inference using only available prior knowledge. The set of samples can then be interrogated for examples that are consistent with any particular data set (a method called *resampling* [13]) or used to parametrise a function that relates data to models which can then be used to solve the inverse or inference problem [14].

✉ Stephanie Earp
stephanie.earp@ed.ac.uk

Andrew Curtis
andrew.curtis@ed.ac.uk

[1] School of Geosciences, University of Edinburgh, Edinburgh, UK

[2] Institute of Geophysics, ETH Zürich, 8092 Zurich, Switzerland

Published online: 06 May 2020

In this work, we use a neural network-based method to perform the inversion. Neural networks (NNs) can approximate any nonlinear relationship between two parameter spaces, given a so-called training set of example pairs of dependent and independent parameter values under that relationship [15]. In travel-time tomography, the forward solution is known and calculable, but the inverse solution is highly nonlinear and nonunique. In such cases, the forward computation can be used to create the prior set of samples known as a *training set*, of random models drawn from the prior pdf; these can be used to train the neural networks to approximate the inverse mapping. The prior samples are only needed during the training process which needs only to be performed once—thereafter, NNs can be evaluated relatively efficiently. This allows the inference step to be run rapidly for any new data set on standard desktop computers, and the overall cost of the method per tomographic problem decreases rapidly with the number of problems to be solved.

Neural network-based inversion methods have been applied to various nonlinear tomography problems in the past. Roth and Tarantola [14] first used NNs to estimate subsurface velocity structure from active source seismic waveforms, Moya and Irikura [16] performed velocity inversion with a neural network using waveform data from earthquakes and Araya-Polo et al. [17] used semblance gathers as input to a network to invert for velocity structure. Gupta et al. [18] used a convolutional network to learn an ensemble of simpler mappings in a low-dimensional space before reconstructing the image by combining the mappings. Dictionary learning methods [19] create sparse representations of the data and can be used to create a set of representations of features. Bianco and Gerstoft [20] performed linearised 2D surface wave travel-time tomography using dictionary learning to regularise the inversion.

The methods mentioned above and in Kong et al. [21] all provide only deterministic solutions to the inversion. Since the solution to tomographic problems is always nonunique, in order to assess the worth of any model estimate, we require that neural networks produce full probabilistic information about the set of models in the inverse problem solution (the posterior pdf). Devilee et al. [11] solved the first probabilistic geophysical inverse problem using NNs. They proposed a variety of methods to train NNs to provide discretised Bayesian tomographic posterior pdfs. Mixture density networks (MDNs) are a class of augmented neural networks that output a probability distribution that is defined as a sum of analytic pdf kernels such as Gaussians [15]. MDNs can be trained such that for any input data this distribution approximates the posterior pdf. These methods have been used to invert surface wave velocities for global crustal thicknesses and seismic velocities [22, 23] and for water content in the mantle transition zone [24], at a reservoir scale to infer petrophysical parameters from velocities [25, 26], for earthquake source parameter estimation [27, 28] and to assess the uncertainty in model parameters of the Earth's global average (one-dimensional) radial velocity structure from *P*-wave travel-time curves [29]. They have also been used in conjunction with Markov random fields and other statistical and graphical models to solve geophysical inverse problems with spatially sophisticated prior information [30–32]. They have been used in conjunction with seismic gradiometry to perform near-real-time 3D surface wave tomography [33]. These studies demonstrate that the pdf obtained from an MDN is comparable to a Monte Carlo sampling solution but is obtained at much lower computational cost in the cases where similar inverse problems must be solved repeatedly with different data sets, and that at the moment of application MDNs provide probabilistic solutions almost instantaneously.

We show for the first time that MDNs can perform fully nonlinear, rapid and probabilistic 2D tomography from travel-time data. We compare different methods for creating the prior training set and performing the neural network inversion. The networks create approximate mean velocity models and estimates of the full marginal posterior pdfs, virtually instantaneously. Thus, in return for accepting approximate posterior pdfs, we obtain a significant computational saving compared to Monte Carlo methods.

2 Method

2.1 Bayesian inference

We wish to solve tomographic inverse problems in a probabilistic framework to find the posterior distribution of velocity models $\mathbf{m}$ that fit some given data $\mathbf{d}$, written as $p(\mathbf{m} \mid \mathbf{d})$. This is defined as [34]:

$$p(\mathbf{m} \mid \mathbf{d}) = k\, p(\mathbf{d} \mid \mathbf{m})\, p(\mathbf{m}) \tag{1}$$

where $p(\mathbf{m})$ represents the prior probability density on the model space, $p(\mathbf{d} \mid \mathbf{m})$ represents the conditional probability of some data given the model (known as the likelihood) and k is a normalisation constant. In multidimensional problems, where the dimensionality of $\mathbf{m}$ is greater than 1, we often need to make inferences about a single parameter with index i and hence must calculate the marginal posterior distribution $p(m^i \mid \mathbf{d})$. This can be obtained by integrating over all parameters j that are not of interest:

$$p(m^i \mid \mathbf{d}) = \int_{\mathbf{m}_j \neq \mathbf{m}_i} p(\mathbf{m} \mid \mathbf{d})\, \mathrm{d}\mathbf{m}_j \tag{2}$$

In this study, we focus on estimating marginal distributions $p(m^i \mid \mathbf{d})$ and posterior trade-offs between pairs of individual parameters.

2.2 Mixture density networks

Neural networks are essentially mathematical mappings that emulate the relationship between two parameter spaces. Given a set of N data–model pairs $\{(\mathbf{d}_i, \mathbf{m}_i) : i = 1, \ldots, N\}$, where $\mathbf{m}_i$ is the model used to generate the data $\mathbf{d}_i$ under some forward relation, NNs can be trained to model an arbitrary nonlinear inverse function from $\mathbf{d}$ to some properties of the set of models $\mathbf{m}$. In this paper, we use a class of neural networks called mixture density networks that can be trained to output the probability of any model $\mathbf{m}$ given some fixed (measured) data $\mathbf{d}$, written as $p(\mathbf{m} \mid \mathbf{d})$. The probability distribution is approximated using a sum (called a mixture) of Gaussians:

$$p(\mathbf{m} \mid \mathbf{d}) \simeq \sum_{i=1}^{M} \alpha_i(\mathbf{d}) \Theta_i(\mathbf{m}|\mathbf{d}) \tag{3}$$

where α_i is called the mixture parameter that attaches relative importance to each Gaussian kernel, M is the number of Gaussians in the mixture and Θ_i are here defined to be Gaussian kernels with a diagonal covariance matrix given by

$$\Theta_i(\mathbf{m} \mid \mathbf{d}) = \frac{1}{\prod_{k=1}^{c} (\sqrt{2\pi}\sigma_{ik}(\mathbf{d}))} \exp\left\{ -\frac{1}{2} \sum_{k=1}^{c} \frac{(\mathbf{m}_i - \mu_{ik}(\mathbf{d}))^2}{\sigma_{ik}^2(\mathbf{d})} \right\} \tag{4}$$

where c is the dimensionality of $\mathbf{m}$, μ_{ik} is the kth element of the ith kernel in the mixture, σ_{ik} is the standard deviation of the kth element of the ith kernel in the mixture and both μ_{ik} and σ_{ik} are outputs of a trained NN. The network is trained by minimising the negative log likelihood of the pdf in Eq. 4, equivalent to maximising the likelihood of the pdf [15]. For a more comprehensive general introduction to MDNs, we refer the reader to Bishop [15], or to Meier et al. [22] and Shahraeeni and Curtis [25] for detailed descriptions with applications in geophysics.

Network training is performed using gradient-based optimisation of the network's internal parameters. The particular trained NN obtained is therefore sensitive to the random parameter initialisation and to the network configuration (internal structure). We train an ensemble of multiple networks with different configurations and combine them to give a group of networks—a so-called *mixture of experts*. In theory, networks trained independently may make good predictions for different reasons and under different inputs (in our case, data vectors); using a combination of networks therefore often results in better generalisation of performance to unseen data and improves prediction accuracy [35]. We construct the ensemble by a weighted average of network outputs, where each weight is determined by the performance of the associated network on the test data set (or simply *test set*). The posterior probability distribution is thus estimated by

$$p(\mathbf{m} \mid \mathbf{d}) \simeq \sum_{i=1}^{M} \sum_{j=1}^{c} \frac{E_i \alpha_{ij}}{\sum_{k=1}^{M} E_k}(\mathbf{d}) \Theta_{ij}(\mathbf{m}|\mathbf{d}) \tag{5}$$

where E_i is the negative exponential of the error on the test data set of the ith kernel. The final estimate of probability distribution $p(\mathbf{m} \mid \mathbf{d})$ contains cM Gaussian kernels.

2.3 Model parametrisation and travel-time data

We define the geometry of our tomography problem to be that shown in Fig. 1. We fix the locations of 18 wave energy sources and receivers (stars shown in Fig. 2) and parametrise the wave speed or velocity across the *Model Volume* within which the forward relationship predicts travel times of the first arriving energy between any source–receiver pair. Travel times $\mathbf{d}_i$ between all possible source–receiver pairs are calculated using an eikonal ray-tracer [36, 37]. The travel times from the four velocity models in Fig. 2 are shown in Fig. 3. Such travel times are used herein to image the velocity structure within the smaller *Image Volume*—wave speeds outside of that area are disregarded and thus constitute nuisance parameters. We use a larger volume to calculate the forward relationship to avoid raypaths travelling along the boundary of the model and causing misleading travel times.

We construct four separate training sets, each of 2.5 million discretised models where each model represents a 2D heterogeneous velocity structure. Two of these training sets are created on an 8×8 coarser grid of cells, and two are created on a 16×16 finer grid of cells within the *Image Volume* (and the same resolution extends throughout the *Model Volume*). Each of the four data sets is created by selecting a random wave speed in each cell independently from the uniform prior distribution U (0.5 km/s, 2.5 km/s). All models in one finer data set and one coarser data set are then smoothed using a 2D averaging filter window which was square of size 5×5 cells for the finer model and 3×3 cells for the coarser model. Thereafter, the velocities are normalised to the same absolute range as the original random models for ease of comparison of results. Then, the travel times between all source–receiver pairs are calculated for all models, in all four training sets (examples are given in Fig. 3).

With this method, we create training sets with two different amounts and types of prior information. The two sets of random unsmoothed velocity models have relatively

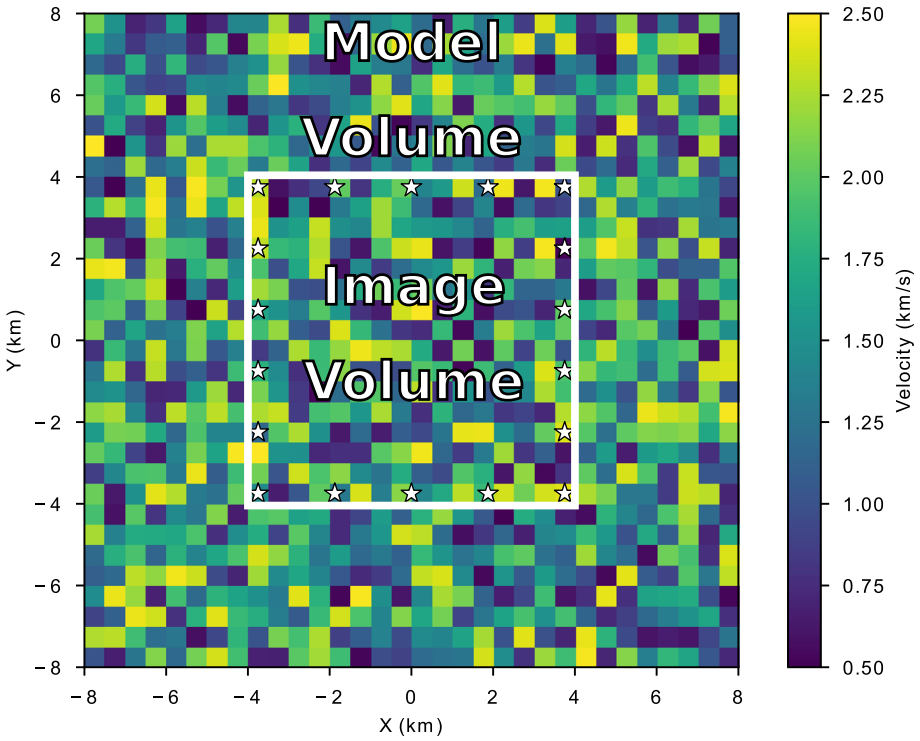

Fig. 1 Geometry of velocity models. Larger model with limits (− 8, 8) in the *X* and *Y* direction is the *Model Volume* within which the travel-times are calculated. The smaller model bounded by a white box with limits (− 4, 4) in the *X* and *Y* direction is the *Image Volume* which we wish to image. White stars represent the location of colocated sources and receivers, between which travel-time data are obtained

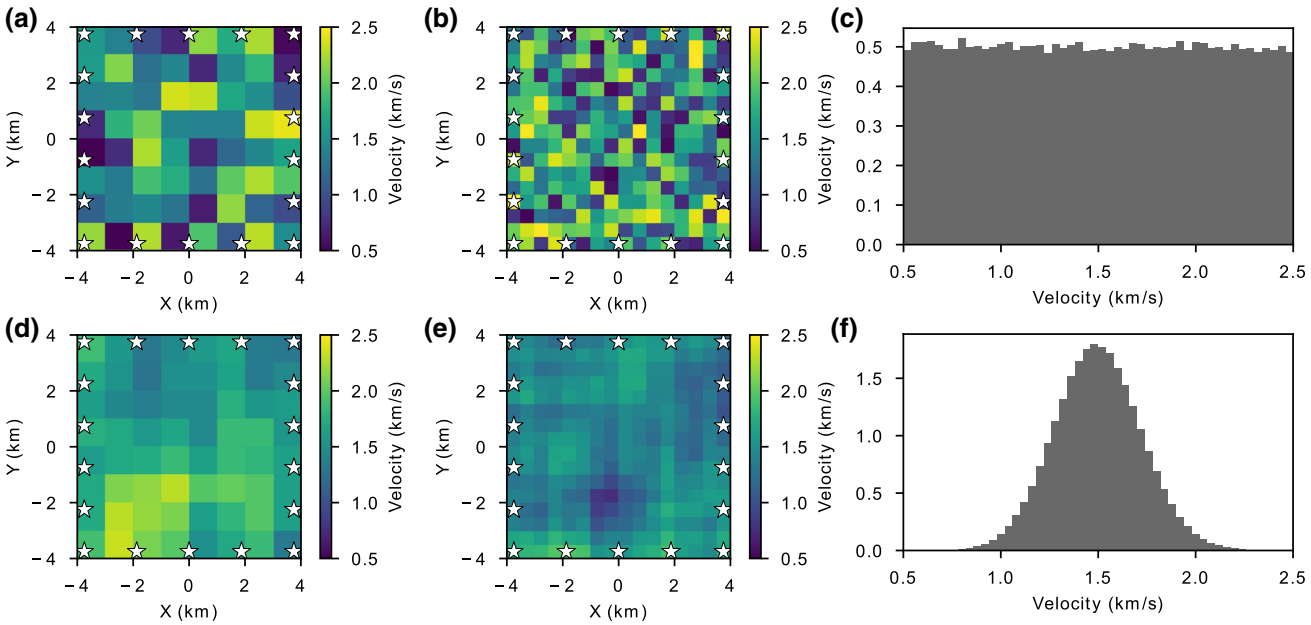

Fig. 2 Example velocity models from the four training sets that are randomly selected from uniform distributions on an **a** 8-by-8 grid and **b** 16-by-16 grid or are randomly selected and then smoothed with a spatial averaging filter on a **d** 8-by-8 grid and **e** 16-by-16 grid. White stars represent the location of colocated sources and receivers. The prior distribution of the training set is shown for one cell in the model given a fixed neighbouring cell for **c** models selected from a uniform random distribution and **f** similar models after spatial smoothing

weak prior information with no correlations between neighbouring cells. This has the advantage that any type of velocity contrast between neighbouring cells would be consistent with the prior pdf and hence can in principle be imaged using the associated trained network given sufficiently informative data (see below). This is demonstrated by the uniform distribution of the histogram in Fig. 2c, which shows the probability of the velocity of the adjacent cell given that the velocity of the central cell is 1.5 km/s. On the other hand, this implies that the prior pdf

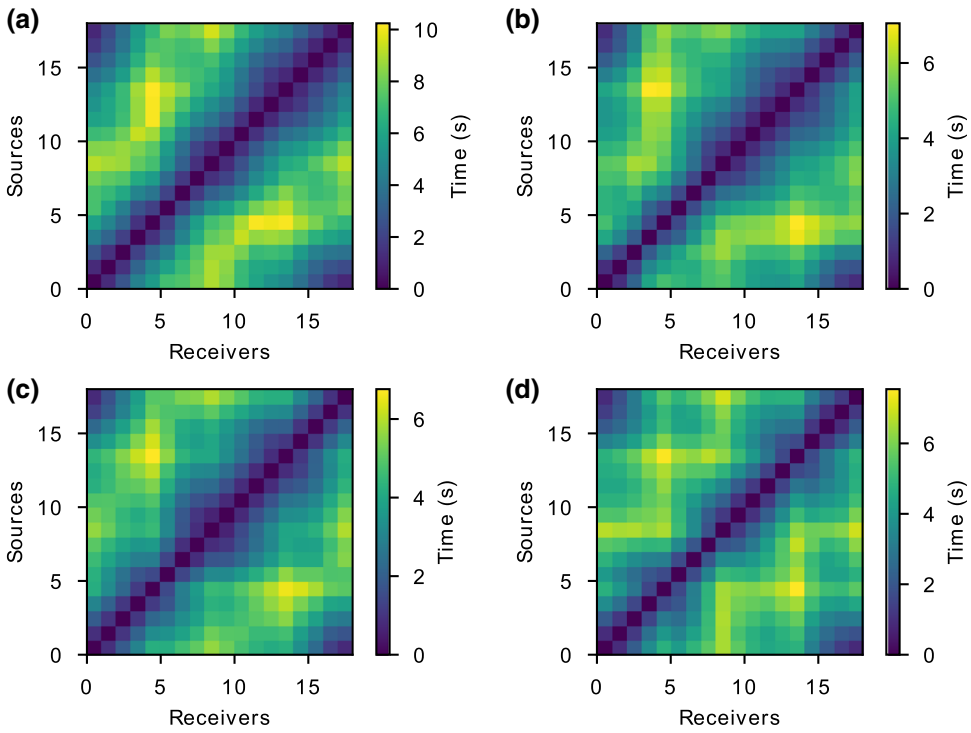

Fig. 3 Corresponding data from the four velocity models in Fig. 2 that are randomly selected from uniform distributions on an **a** 8-by-8 grid and **b** 16-by-16 grid or are randomly selected and then smoothed with an averaging filter on a **c** 8-by-8 grid and **d** 16-by-16 grid

is uniform over a 64- and 256-dimensional space for the coarser and finer training sets, respectively; these spaces are therefore extremely sparsely sampled by the 2,500,000 training set models due to the curse of dimensionality [38]. This implies that over most of these two spaces the prior pdf is entirely unrepresented by "proximal" samples.

The two sets of smoothed velocity models embody stronger prior information as the speeds in neighbouring cells are correlated. This is demonstrated in Fig. 2f where the distribution of possible velocities in adjacent cells given that the velocity of the central cell is 1.5 km/s is approximately Gaussian. This means that models with larger velocity contrasts between neighbouring cells are not represented in the training data set and hence will be precluded from inversion results. This may or may not be advantageous depending on the true prior information about the form of the structure being imaged. However, it has the advantage that the effective space (manifold) of models consistent with the prior information is considerably smaller than that for the smoothed models, so that the finite-sized training set may better represent the form of the prior pdf.

3 Results

3.1 Network configurations

We train separate MDNs to predict the marginal probability distribution $p(m^i \mid \mathbf{d})$ of velocity m^i in cell i in each of the two sizes of models. For the finer data sets, we train four MDNs and for the coarser data sets we train eight MDNs at each location i. We use different configurations as well as randomly initialised internal network parameters (commonly referred to as weights and biases) for each network because diversity in the ensemble generally leads to better predictions [35]. "Appendix 1" outlines the different network configurations. For each network, we use a Gaussian mixture consisting of 15 kernels. The precise number of kernels is not important as long as it is larger than the number required to represent the marginal posterior pdf in each model cell. The network can either reduce the amplitude of the mixture parameter α_{ij} to close to zero to remove unnecessary kernels or can combine unnecessary kernels by giving them a similar μ and σ to other kernels [15]. In practice, we found the maximum number of kernels with significant weight used in any mixture was 8.

We also train networks to invert for the full model (velocities in all cells at once) using a single network. In this case, we use a convolutional network with three convolutional layers followed by three fully connected layers and 15 kernels for the Gaussian mixture. We train ten networks with five different network configurations (each

configuration is trained twice with random weight initialisation). Layer sizes were selected using the Python library hyperopt [39], and "Appendix 1" gives further description of the networks used. The same network configurations were trained on all four training sets.

For every training run for each network configuration, we use 85% of the training data set to train the network, 10% of the data set as a validation set during training and 5% as a test set to evaluate the final network once training has finished. The training set is used in the optimisation of network parameters. The parameters are updated iteratively so that the output of the network best represents the training set sample distribution. To avoid overfitting the network to the data, the cost function is also periodically evaluated over the validation set; when the error on the validation set stops decreasing, we end the training optimisation. Once all of the networks have been trained, we evaluate the final network performance using the test set and sum the networks across the ensemble using Eq. 5.

3.2 Result evaluation

We tested our trained networks by applying them to synthetic data sets calculated for velocity models created specifically to test the performance of each type of network. The quality of the mean of the inverted probability distributions of 2D velocity models (comprising 1D marginal posterior pdfs in each model cell in the cases where networks were trained for each cell individually) is compared against the true velocity model using the structural similarity index metric (SSIM). This metric is based on three relatively independent comparison measurements: luminance, contrast and structure ("Appendix 2"). SSIM can assume values between -1 and 1: a value of 1 indicates the images are identical, 0 indicates no structural similarity and negative values occur when local structure is inverted. SSIM differs from other quality indicators such as mean squared error (MSE) in that it measures the quality of an image in structure and pixel value compared to a ground truth, rather than the absolute squared errors (which often do not mean much to someone who is trying to interpret the resulting images).

We compare the information gain between the prior $p(\mathbf{m})$ and the posterior $p(\mathbf{m}|\mathbf{d})$ distribution using the Kullback–Leibler (KL) divergence

$$D_{\mathrm{KL}}(p(\mathbf{m}|\mathbf{d}), p(\mathbf{m})) = \int_{-\infty}^{\infty} p(\mathbf{m}|\mathbf{d}) ln\left(\frac{p(\mathbf{m}|\mathbf{d})}{p(\mathbf{m})}\right) \mathrm{d}x \qquad (6)$$

where a higher D_{KL} indicates that the posterior pdf has gained information over the prior and $D_{\mathrm{KL}} = 0$ occurs when the two distributions are the same. This can be used as an indication of the effectiveness of the network: if D_{KL} is close to 0, then the network has been able to learn little, if anything at all, from the data.

3.3 Prior

To show the effect of the prior on our models, we inverted synthetic data for the three velocity models shown in Figs. 4a and 5a using networks trained with weak prior information (unsmoothed training models) in Fig. 4 and those trained with stronger prior information (smoothed models) in Fig. 5. The test models were defined on a 32×32 grid, which is finer than either of our training sets; this ensures that we evaluate the networks using models that are outside the range of those used for training. For all test models, it is clear that with stronger prior information, the networks better resolve the velocity structure, shown generally by the much higher SSIM values in Fig. 5b, c compared with the corresponding values in Fig. 4b, c. This is true even though the test models contradict the stronger prior information: they all contain structures that at least in part are not smooth.

The velocity model in the left-hand column has a background velocity (cells surrounding the central anomaly) equal to the mean of the prior pdf and a circular low velocity and is estimated well in both inversions using weaker prior information training sets (Fig. 4). However, even a small increase in complexity in velocity models gives poor inversion results as shown by the central column of velocity models. For these, all the velocities are increased compared to the left column, and in particular the background velocity is increased away from the mean of the prior. In this case, the networks with weaker prior information are unable to recover much, if any, of the true structure. If stronger prior information is included in the training set, the networks accurately predict a larger variety of velocity models. The true structures of the two circular models in Fig. 5 are closely reproduced in the inversion. Sharp contrasts in velocity in the true model are translated to more gradual changes in velocity in the estimates (for both grid sizes) due to the smoothness in the prior pdf. Despite this, the SSIM values show that results are very well correlated with the true model. For the more geologically reasonable model in the right column of Fig. 5 which includes a structure that might be generated by a fault, networks trained using stronger prior information on both grid sizes produce models that are nearly identical to the true model. Even though the true model contains a sharp contrast boundary, the inverted models still contain a (slightly smoother) version and the overall structure of the true image is maintained.

The effect of stronger prior information is shown in the posterior pdfs in Fig. 6. We display the posterior marginal pdfs at three locations indicated in the upper right-hand

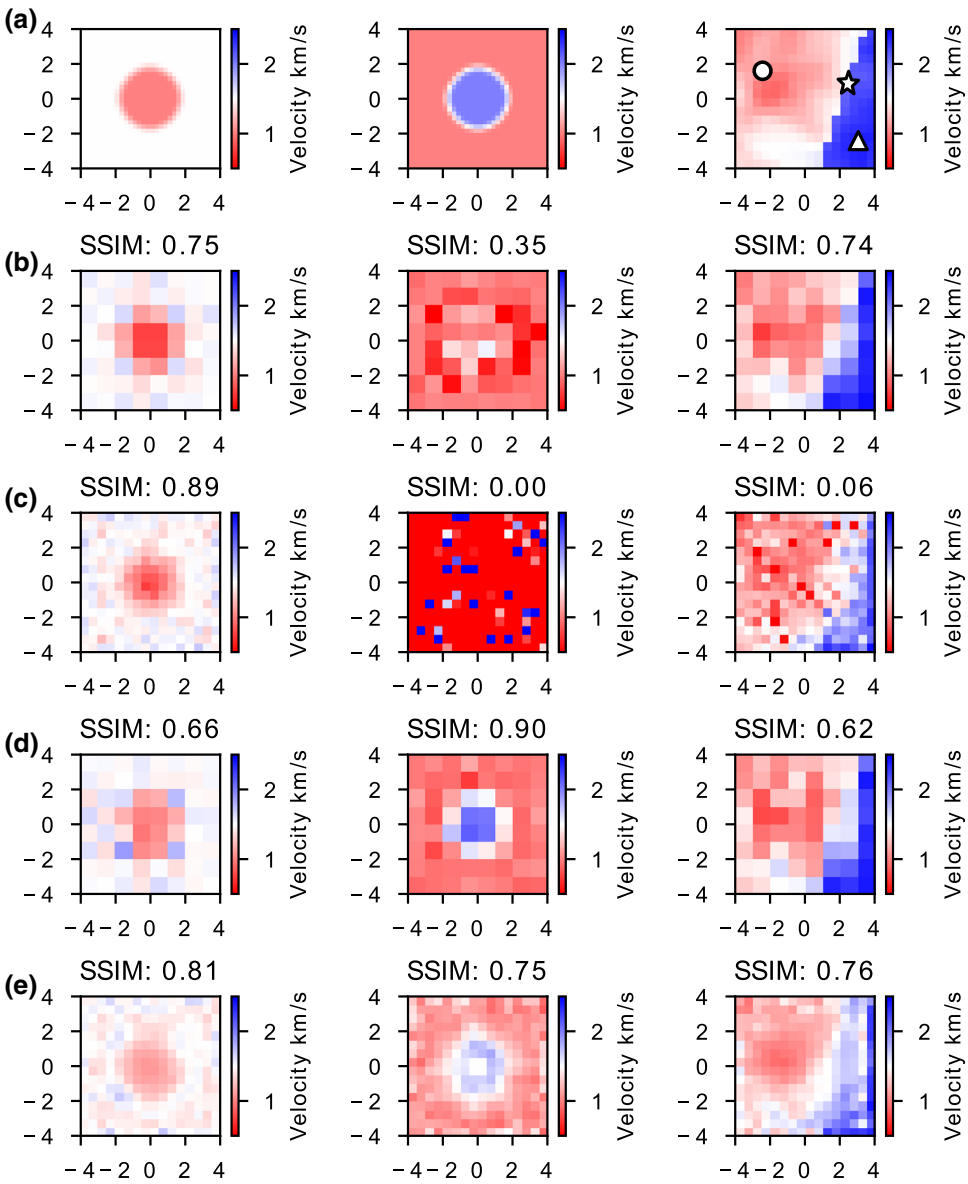

Fig. 4 **a** True velocity models. Using a randomly generated training set drawn from a uniform distribution, mean velocities from single-cell MDN inversions for **b** an 8×8 model and **c** a 16×16 model, and from full-model MDN inversions for **d** an 8×8 model and **e** a 16×16 model. The corresponding SSIM values are shown above each result (see "Appendix 2" for definition of SSIM)

model in Fig. 4a: a location in the high-velocity zone (triangle), the low-velocity zone (circle) and at the edge of the sharp contrast where the inversion struggles to image correctly (star). The KL divergence values are shown above the corresponding posterior marginal pdf. The most striking feature is the much higher KL values for the networks trained with the stronger prior information (rows b and d) indicating a larger information gain in the posterior pdf compared to the prior pdf than is obtained when training with uniformly random models. In fact, the low KL values for the latter cases imply that nearly no information was gained from the data, and even though a rough approximation of the mean can be found, the uncertainties on those values remain large.

3.4 Model resolution

Our networks are trained on two sizes of grid cell, a coarser 8×8 grid and a finer 16×16 grid. Figures 4 and 5 show the results for varying grid sizes. Training on the finer grid induces a factor of four more parameters to estimate from the same data. This means that a larger training set size would be needed to sample the increase in image dimensionality. It would be impossible to sample densely the 256-dimensional space spanned by a 16×16 grid, but as our examples show, the networks are still able to invert for some basic structural information (Fig. 4c). When we train our networks with a stronger prior pdf, we reduce the effective dimensionality of our problem by introducing a relationship between neighbouring pixels: essentially all prior models and hence most posterior models lie on a

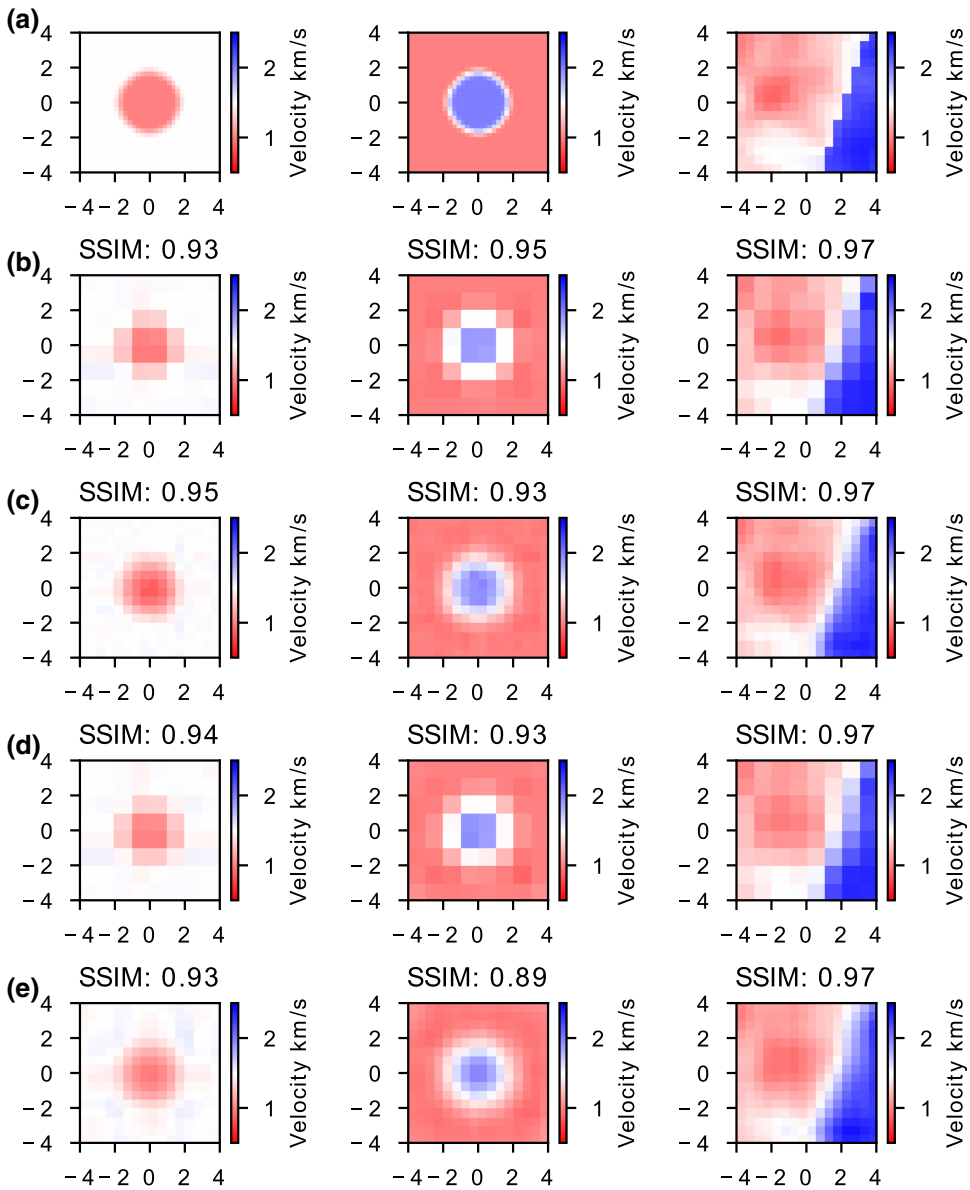

Fig. 5 **a** True velocity models. Using a training set with spatially smoothed velocities, mean velocities from single-cell MDN inversions for **b** an 8×8 model and **c** a 16×16 model and from full-model MDN inversions for **d** an 8×8 model and **e** a 16×16 model. The corresponding SSIM values are shown above each result (see "Appendix 2" for definition of SSIM)

significantly lower dimensional manifold that is embedded with the 64- or 256-dimensional spaces. In that case, we can obtain reasonable estimates of the true velocity models regardless of grid size (Fig. 5c).

3.5 Type of network

For each of the four training sets, we trained networks in two different ways. First, we trained separate networks to estimate marginal pdfs in each cell so that each network has fewer parameters (α_{ij}, μ_{ij}, σ_{ij}) to estimate. Note that this does not reduce the dimensionality of the overall problem as each velocity cell in the model contributes to the travel-time values, and the velocity in any cell depends on the cells surrounding it even if we do not directly invert for them within the same network. It is important to remember that in this case we do not obtain explicit information about trade-offs between neighbouring cells. Those trade-offs are already integrated into the marginal pdfs in Eq. 2.

We also trained networks to invert for slowness in every cell of the model at once. This increases the number of parameters that the network must estimate but as a result the trade-off between velocity values in adjacent cells can be explored. Examples of the joint marginal pdfs from the central model in Fig. 4a are shown in Fig. 7: the 2D pdfs show few signs of nonlinearity, and virtually no indication of the trade-offs that one would expect between velocities in neighbouring cells. This indicates that the results of these networks are unlikely to provide reliable uncertainties.

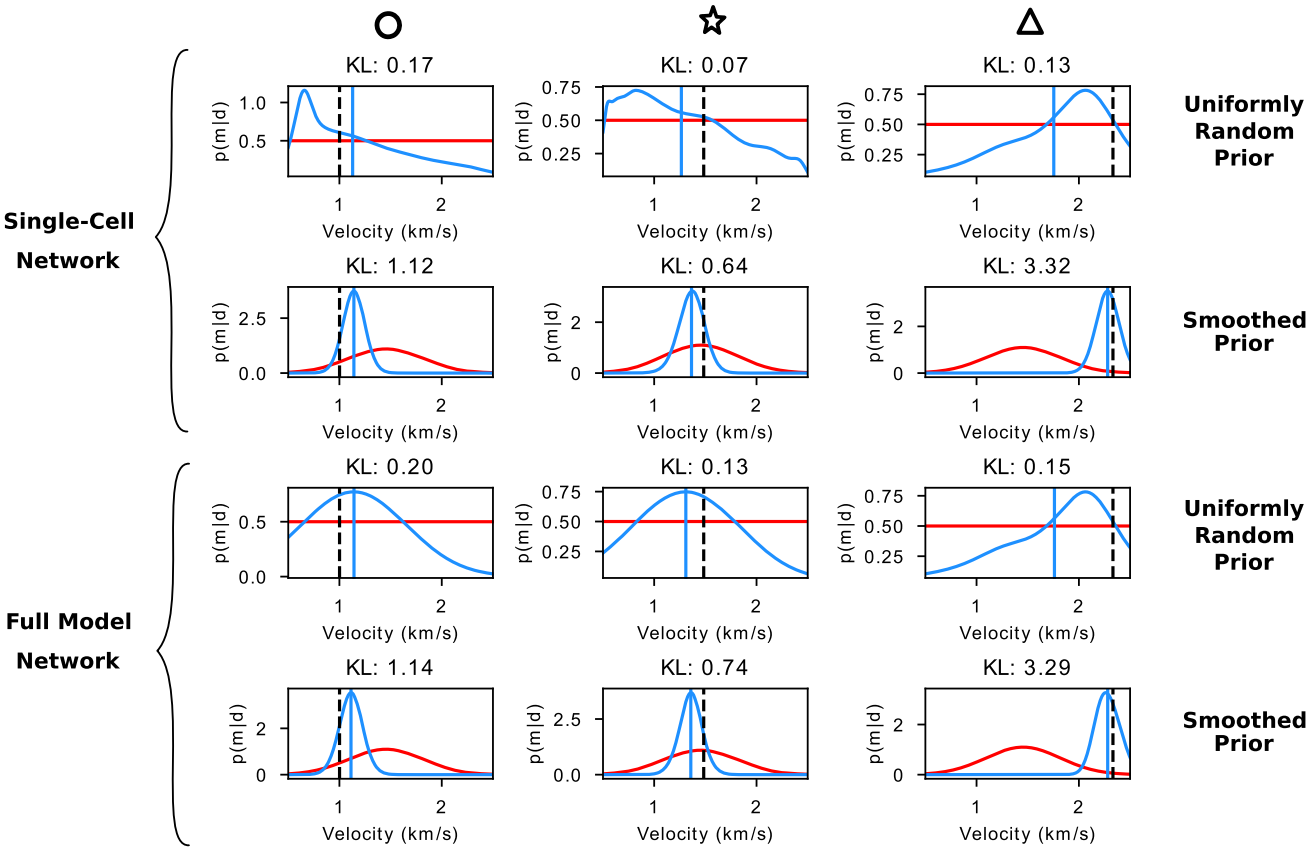

Fig. 6 Posterior pdfs (blue curves) compared to the prior pdfs (red curves) for the 16×16 grid models for three locations shown in the top-right model of Fig. 4: circle (left), star (middle), triangle (right). The rows show results from: (row 1) single-cell MDN's using uniformly random training data set. (Row 2) Single-cell MDN's using the smoothed training data set. (Row 3) Full-model MDN using uniformly random training data set. (Row 4) Full-model MDN using the smoothed training data set. The mean of the posterior is shown by the blue solid line and the true velocity value by a black dashed line. Corresponding KL divergence values are shown above each result (colour figure online)

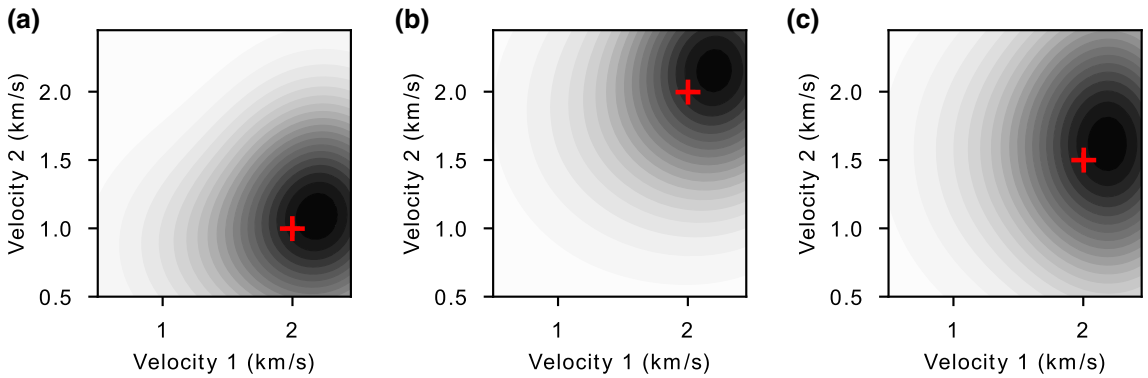

Fig. 7 Joint pdfs comparing the pixel inside the velocity high of the central model in Fig. 4a. Velocity 1 is the velocity of a cell in the centre of the velocity high. Velocity 2 is the velocity of a cell **a** in the background velocity, **b** at the centre of the velocity high (not the same cell as Velocity 1) and **c** at the edge of the velocity anomaly

For models on a coarser grid (Figs. 4 and 5 rows b and d), networks perform similarly when using the single-cell networks or the full-model networks. For models trained on a finer grid, the full-model networks perform significantly better than the single-cell network as shown in Fig. 4. This is almost certainly because the dimensionality of the problem when training single-cell networks is too large, but by giving the full-model network information about the velocities in neighbouring cells, it can better resolve the

velocities. This difference is less noticeable when using stronger prior information (Fig. 5b, d).

3.6 Uncertainty loops

A key problem in the field of nonlinear inversion is that there are no standard solutions to which estimated posterior pdfs can be compared in order to verify their quality. In almost all papers that use synthetic tests to assess competing methodologies in high-dimensional problems, the main criterion applied is whether the mean or maximum-likelihood model fits the real (true synthetic) model that was used to generate the synthetic data. This provides no test at all on the rest of the pdf, and indeed, there is no reason why the mean *should* match the true model in nonlinear problems—the mean may even be a zero-probability solution (one precluded by the data) [34]. The maximum likelihood (or maximum posterior probability) model is an alternative, but is usually an extremely volatile statistic of pdf solutions since those solutions are necessarily formed by focusing across the whole pdf rather than simply on its modes. We therefore require some independent property of posterior pdfs, the existence of which we can use to assess their veracity.

Loops or halos of high uncertainty have been shown to exist in solutions to all travel-time tomography problems around anomalies with a spatially sharp and strong contrast in velocity compared to their surroundings [5]. Uncertainty loops exist due to nonlinear aspects of wave physics and represent uncertainty in the *shape* of such anomalies. They are observed most clearly in fully nonlinearised tomographic inversion problems in which rays, velocities and travel times are all varied in concert for each sample considered. We can therefore use the existence of loops in posterior uncertainty as a criterion to check their quality in models with strong and spatially sharp contrasts.

Figure 8 shows the standard deviations (bottom row) for the results of networks trained on an 8×8 grid. Only the networks trained using the training set of uniformly random velocities (Fig. 8f, h) exhibit signs of an uncertainty loop. We include the mean (middle row) for comparison of the shape of the velocity anomaly to the loop that surrounds it. The difference between the two priors is clear when comparing Fig. 8f–h: for a smoothed prior (Fig. 8g), the maximum uncertainty is predicted to be in the centre of the anomaly as opposed to the other two images where the uncertainty is lowest at the centre of the anomaly and highest on the margins as expected. However, when inverting for the full model in a single network (Fig. 8h), the loop is not as well defined as in Fig. 8f. Together with the lack of clear trade-off relations in Fig. 7, this is evidence that the full-model inversions are less robust than single-cell inversions: as the networks invert for many more parameters at once, they appear not to have been trained so as to fully represent the correct physics of the tomography problem.

The single-cell networks (one network trained for each cell in the velocity model) allow us to estimate the full marginal posterior probabilities for all cells in the model, and these posterior distributions show how the network represents uncertainty. We show the pdfs for three points in the model: inside the velocity anomaly (star), at the edge of the anomaly (triangle) and in the background velocity (circle), where the locations are shown in Fig. 8a. We can see for the 8×8 model using the uniformly random training set (Fig. 9a, c) the posterior pdf at the edge of the anomaly has a larger uncertainty indicating that the range of possible velocities spans the velocity of the anomaly and that of the background velocity. This is expected at the edge of an anomaly, the boundaries of which are uncertain: the cells could either be inside or outside of the anomaly and could therefore assume values of the anomaly (low velocity) or the background model (high velocity). This is the maximum range of velocities expected across the model; hence, the largest uncertainties should be around anomaly edges [5].

We do not see uncertainty loops in any model trained on the smoothed models. This makes sense because by imposing prior information that the model is relatively smooth, we have removed the possibility to include the effect of spatially sharp contrasts between anomalies and the background velocity model, precluding the types of physical trade-offs that create uncertainty loops. This is represented in the pdfs (Fig. 9b, d) where the uncertainty is much smaller than in (a) and (e) and where there is no noticeable increase in uncertainties at the boundary of the anomaly. Note that there is again a larger information gain for the results from the smooth training set as shown by the KL divergence values.

3.7 Realistic velocity models

Figures 10 and 11 show the results when applying the trained networks to other types of structures that might be encountered in geophysical or nondestructive testing applications. Figure 10 shows results using uniformly random training set, whereas Fig. 11 shows the equivalent results obtained using the smoothed training set. The models inverted on a coarser grid produce reasonable estimates of the velocity models using either prior pdf; however, for the smoothed prior, all the models, regardless of grid size, are recovered fairly well. Figure 12 shows the uncertainty maps for a coarse grid model trained using both types of prior information and inverted using the single-cell MDN models. When inverting the models with a uniformly random prior (Fig. 12b), the uncertainty maps show a

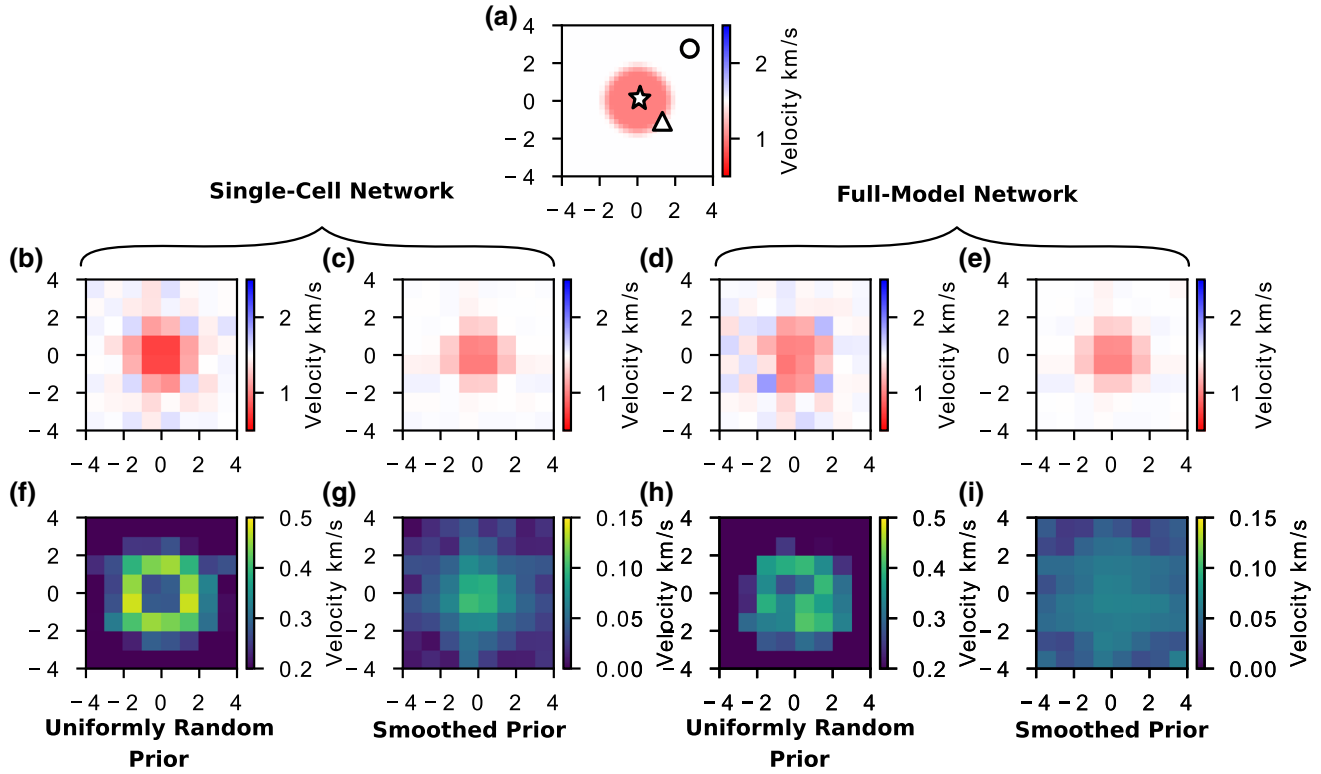

Fig. 8 **a** True velocity model. For a single-cell MDN, using a training set from a uniformly random distribution, results shown are **b** mean velocities and **f** corresponding standard deviations. Using the same type of network with a training set of spatially smoothed velocities, we obtain **c** mean velocities and **g** standard deviations. For a full-model MDN, using a training set from a uniformly random distribution, we obtain **d** mean velocities and **h** standard deviations. Using the same type of network with a training set of smoothed velocities, we obtain **e** mean velocities and **i** standard deviations

higher uncertainty at the anomaly interfaces (as expected by analogy with the uncertainty loops above), thus helping to define uncertainty in the model geometry, while the results from the smooth prior miss this extra information.

4 Discussion

We compared different methods of mixture density network inversions to estimate tomographic posterior probability density functions. When using data sets with little prior information (Fig. 2a, b), the networks struggle to estimate more than the simplest of velocity models: due to the curse of dimensionality, it is simply not possible to provide a sufficient density of prior samples on which to train the MDN. Including stronger prior information in our examples by training on smoothed velocity models (Fig. 2d, e) improves inversion results, although the networks are no longer able to image sharp velocity contrasts, nor estimate uncertainty in the shapes and locations of spatially sharp velocity anomalies, as information about such models is not contained in the training set. Our tests indicate that the prior pdf is the most important factor in improving a network performance since it restricts both the training set and inversion results to a more constrained (effectively lower-dimensional) manifold embedded within the high-dimensional parameter space. This manifold is more densely sampled than the full space, thus improving network training and performance. All test models inverted using the stronger prior information give higher SSIM and KL divergence values compared to those using weaker priors, regardless of grid size or how many pixels were inverted with each network. Also, the two circular anomalies in Fig. 5 are symmetrical, and this symmetry is also shown in all of the smooth-prior inversion results which is not seen in the uniform-prior results in Fig. 4. Nevertheless, we show that when imposed prior information is false (if the true model is rough but the prior precludes such models), then uncertainty results will be compromised as in Fig. 8g, i. It should be noted that neither of the training sets created in this study are fully representative of the true Earth. In reality, actual geophysical features of the Earth are neither uniformly random nor smoothed, but are dependent on geological characteristics that can include smooth variations or sharp boundaries. The results in Figs. 10 and 11 show that different structures not seen in the training set can be recovered using this neural networks method. However, a clearly

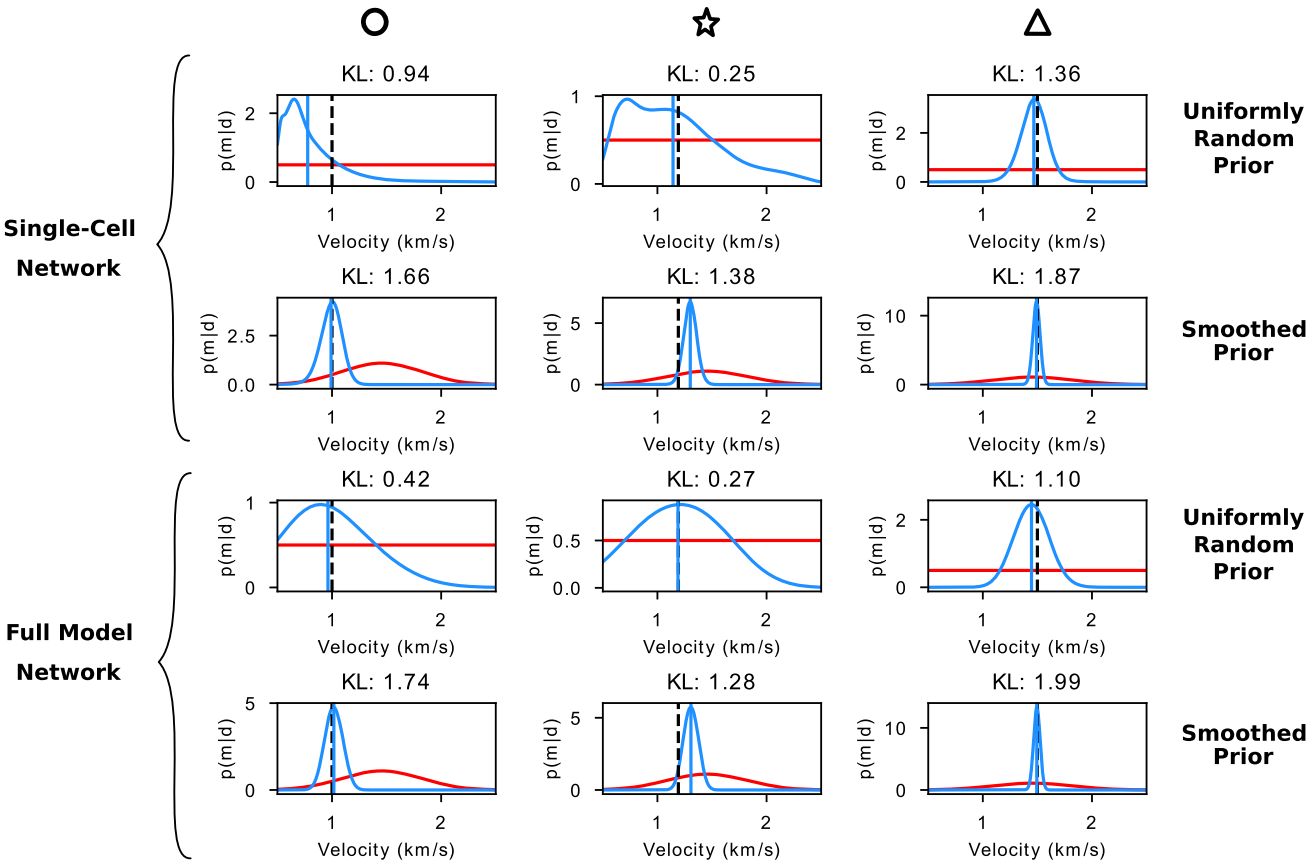

Fig. 9 Posterior pdfs (blue curves) compared to the prior pdfs (red curves) for the 16×16 grid models for three locations shown in the true model of Fig. 8: circle (left), star (middle), triangle (right). The rows show results from: (Row 1) Single-cell MDN's using uniformly random training data set. (Row 2) Single-cell MDN's using the smoothed training data set. (Row 3) Full-model MDN using uniformly random training data set. (Row 4) Full-model MDN using the smoothed training data set. The mean of the posterior is shown by the blue solid line and the true velocity value by a black dashed line. Corresponding KL divergence values are shown above each result (colour figure online)

advantageous strategy for the future of neural network tomography is to invest effort in finding and using more sophisticated and correct prior information [40]. Recent efforts in this direction include [41] who use expert elicitation to constrain prior multipoint geostatistics, Mosser et al. [42] who use neural networks to parametrise geological prior information and Nawaz and Curtis [30–32] who use Markovian models and variational methods with embedded neural and mixture density networks to combine geological and geophysical information; these various directions appear to be strategically important for the future of this field.

We illustrate the differences in the KL divergence values in Fig. 13. The top graph shows histograms of KL values obtained when networks are applied to all synthetic test data for the four different prior and network training types for the 8×8 grid model, and the bottom graph is similar but for 16×16 models. Both plots confirm that training with a stronger prior increases the information gain in the posterior as indicated in Fig. 6. Notice that this is not necessarily an intuitively obvious result: if prior information is weaker or less informative, we might expect the data to add relatively more information, compared to the case where prior information is stronger. We therefore suspect that this result indicates that we simply cannot train the MDNs in the case of weaker prior information and sparser training examples; even though by adding stronger prior information, we should *decrease* the relative value of the data, this effect is outweighed by the fact that we can better train the network and thereby extract *more* information from data.

The effect of increasing the number of cells in the model is also clearly highlighted: Fig. 13a has higher KL values than Fig. 13b. Interestingly, both plots show that training using a full-model inversion slightly increases the KL divergence, implying that the networks are making use of the relationship between adjacent pixels to better constrain the posterior pdfs.

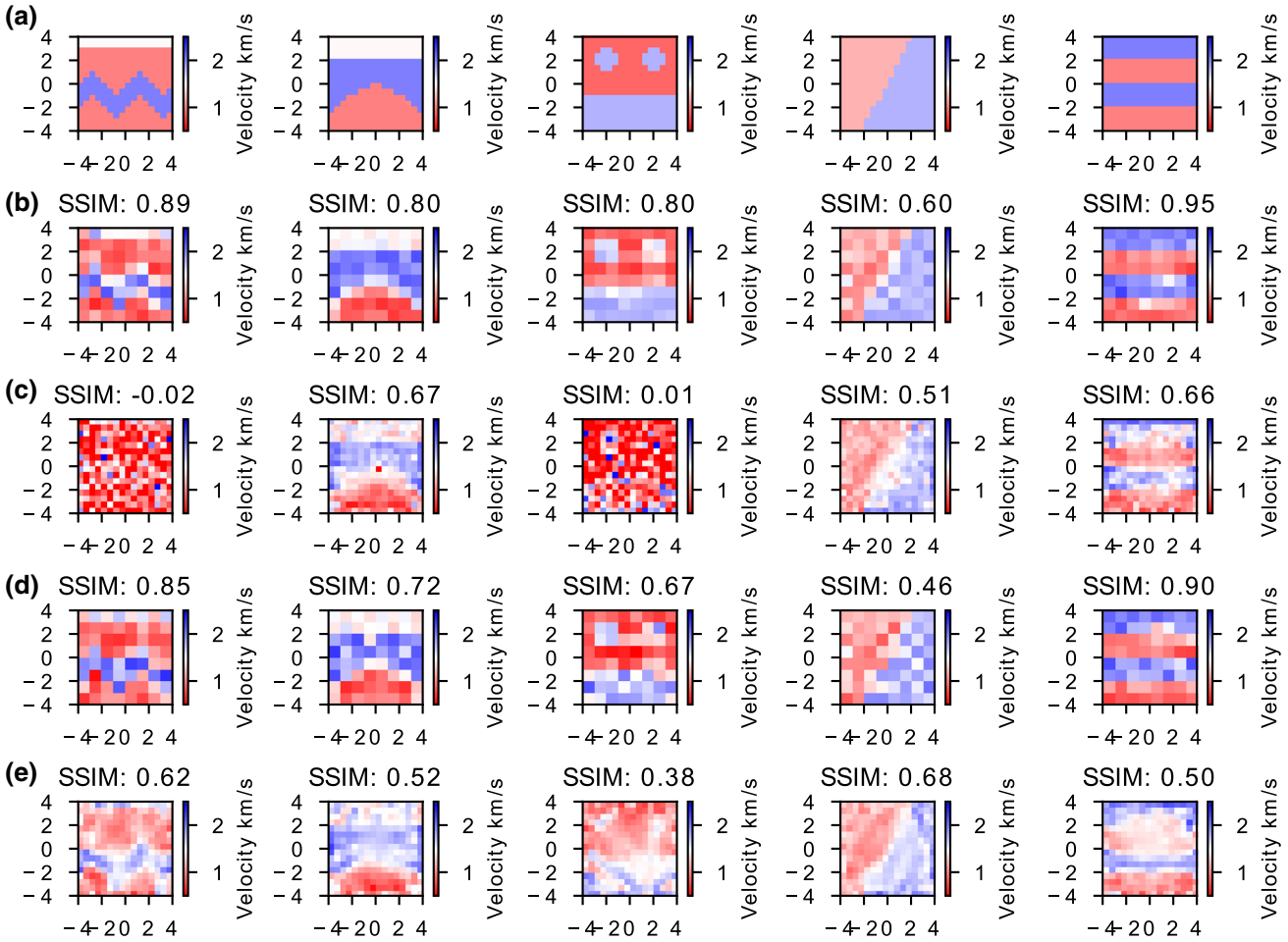

Fig. 10 **a** True velocity models. Using a randomly generated training set from a uniform distribution, mean velocities from single-cell MDN inversions for **b** an 8×8 model and **c** a 16×16 model and from full-model MDN inversion for **d** an 8×8 model and **e** a 16×16 model. The corresponding SSIM values are shown above each result (see "Appendix 2" for definition of SSIM)

4.1 Inference limits

When creating the training data set, we set hard bounds on the grid cell velocities, thus limiting the range of velocity models that should be found using the trained networks. Figure 14 shows the inversion of a model at the limits of all training data sets. The middle row shows results when using the uniformly random training data set: none of the inversions give reliable results. Although the network trained to invert the full model at once performs slightly better, all networks produce extremely poor results. This is expected as the velocity model has a background velocity at the lower limit of the training sets, 0.5 km/s and an anomaly at the upper limit, 2.5 km/s. This is an extreme example that is not likely to have proximal samples in the training set; therefore, the results are expected to be poor.

The same model lies out with the data set with a stronger prior as well, but networks appear to recognise that there is a velocity anomaly. However, since the prior data set used is smoothed, strong contrasts are precluded and none of the networks give accurate velocity information, despite being able to represent the geometry of the structure.

4.2 Inversion speed

As this is a prior sampling method, the training data set must be created in advance. It took $t_{\text{prior}} = 11$ h, to create the training data set of 2.5 million samples using 5 CPUs on a Dell PowerEdge R820. However, this only needs to be done once; even if more prior information becomes available, we may be able to update our prior using the *prior replacement* method of Walker and Curtis [43] or the resampling method of Sambridge [13] rather than calculating entirely new training examples.

In this work, each network took between 1 and 2 h to train (converge) using 16 GB of RAM over 2 NVIDIA TITAN X GPUs. For the 8×8 grid models with an ensemble of eight networks when training the network for

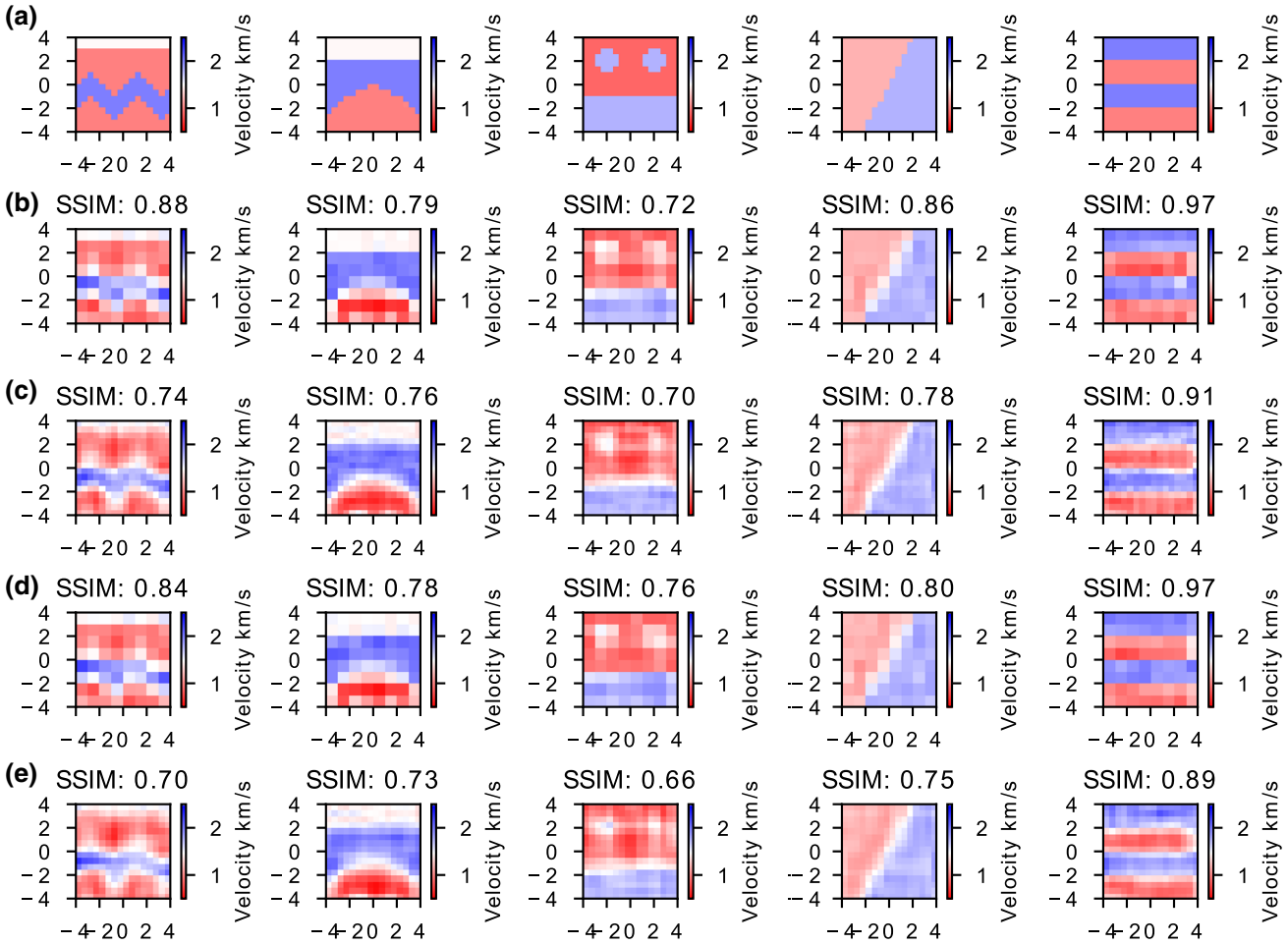

Fig. 11 **a** True velocity models. Using a training set drawn of smoothed random models, mean velocities from single-cell MDN inversions for **b** an 8×8 model and **c** a 16×16 model and from full-model MDN inversions for **d** an 8×8 model and **e** a 16×16 model. The corresponding SSIM values are shown above each result (see "Appendix 2" for definition of SSIM)

each grid cell separately, we required $8 \times 8 \times 8 = 512$ networks in total and for the 16×16 models with an ensemble of four networks, we required $16 \times 16 \times 4 = 2048$ networks. However, the training of each network is independent of others so the process can easily be parallelised and using 50 cores a full training run for the larger 16×16 grid model takes $t_{\text{train}} = 80$ h of real clock time. For the full-model networks, only one network is trained for all cells so the total training time is much lower: each network takes around 3 h to train using 48 GB of RAM over 4 NVIDIA TITAN X GPUs so training ten networks only takes 30 h without running them in parallel. This process could be reduced to 3 h by using only ten cores and reduced further by training each network across cores. The advantage of an MDN is the speed of inversion after training: once a network is trained, new inversions take a fraction of a second, even on a standard desktop computer. Computational efficiency is therefore gained only when the trained networks will be applied to many different data sets.

Monte Carlo methods are known to be computationally expensive [6], and a fully nonlinear Markov chain Monte Carlo (McMC) tomographic inversion can take weeks or months of compute time. Monte Carlo methods use posterior sampling so for every new inversion a new sample set must be performed. This is often a far less demanding sampling task than sampling with similar density of samples from the prior since high-probability parts of the posterior pdf usually span a significantly smaller volume of parameter space. Nevertheless, neural network methods are advantageous over traditional Monte Carlo methods when n repeated inversions of similar data types are to be performed provided that $n > \frac{(t_{\text{prior}} + t_{\text{train}})}{t_{\text{MC}}}$, as the computationally expensive sampling step only needs to be performed once and the network-based inversion becomes faster. In a tomographic setting, this could be useful for monitoring purposes, where data collected periodically from the same

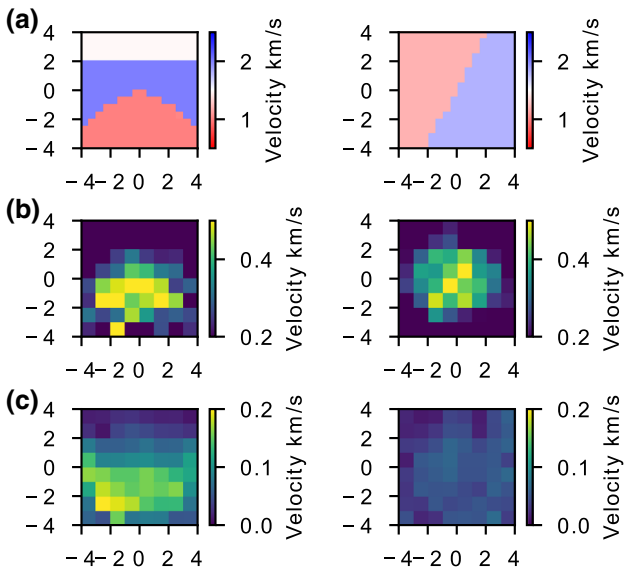

Fig. 12 **a** True velocity models. For a single-cell MDN, **b** the standard deviations using a training set generated from a uniformly random distribution. Using the same network with a training set of smoothed velocities, we obtain standard deviations (**c**)

set of sources and receivers can be inverted with the same network(s) each time new data arrive. However it should be noted that despite the longer computational time, Monte Carlo methods can be used to produce higher resolution 2D or 3D models [10, 44]. Mixture density networks have also been shown to give conservative uncertainty estimates compared to Monte Carlo methods [12].

4.3 Training flexibility

In this work, we train networks assuming that the data (travel times) are recorded with exactly the same data acquisition geometry as was used for training. It would also be possible to train more flexible networks that account for missing data. For example, one could augment the training set with additional samples constructed from the same data–model pairs $(\mathbf{d}_i, \mathbf{m}_i) : i = 1, \ldots, N$ but with a certain number of travel-time values in the data set randomly set to 0, to indicate a missing value [29]. Then, new data sets with a missing values (for example due to a noisy stations causing errors in travel times) can be inverted by the same network.

Data from new receivers added after training the network cannot be used. However, we can create a new training set containing only the data from the added receiver station and fine-tune the original network by using the original network parameter values as a starting point for training optimisation. This has the advantage that the training process will be much faster.

5 Conclusion

We present neural network-based, nonlinear inversion methods applied to a 2D travel-time tomography problem to estimate posterior probability density functions. The flexibility of mixture density networks means that we can provide uncertainty estimates for 2D velocity maps. We show that the prior information used to create the training data set is the most important factor in providing accurate

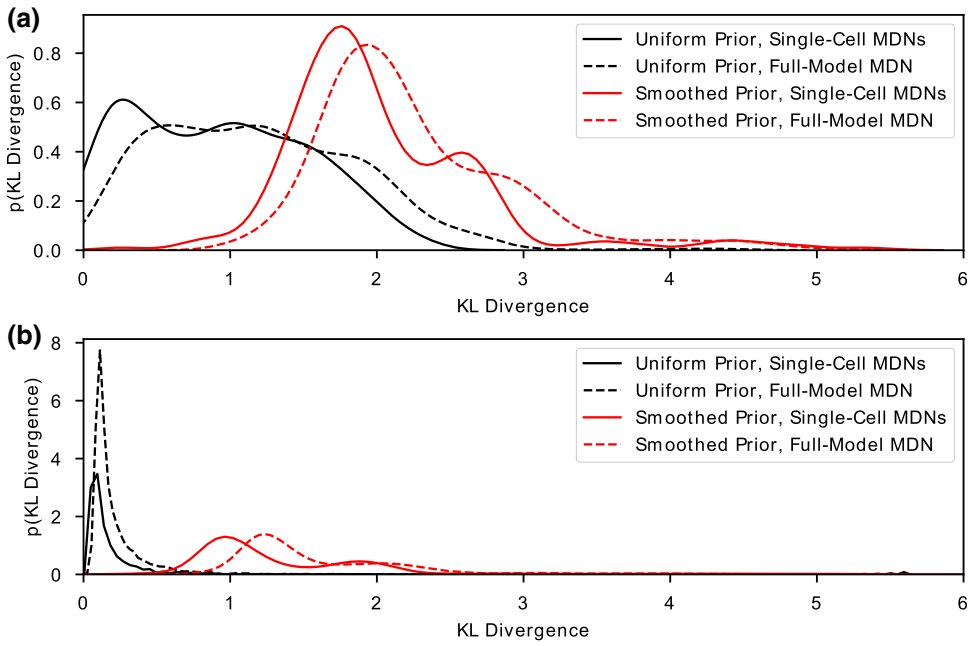

Fig. 13 Histograms of KL divergence values for results of inverting synthetic data for all models in the test set. **a** 8×8 models, **b** 16×16 models

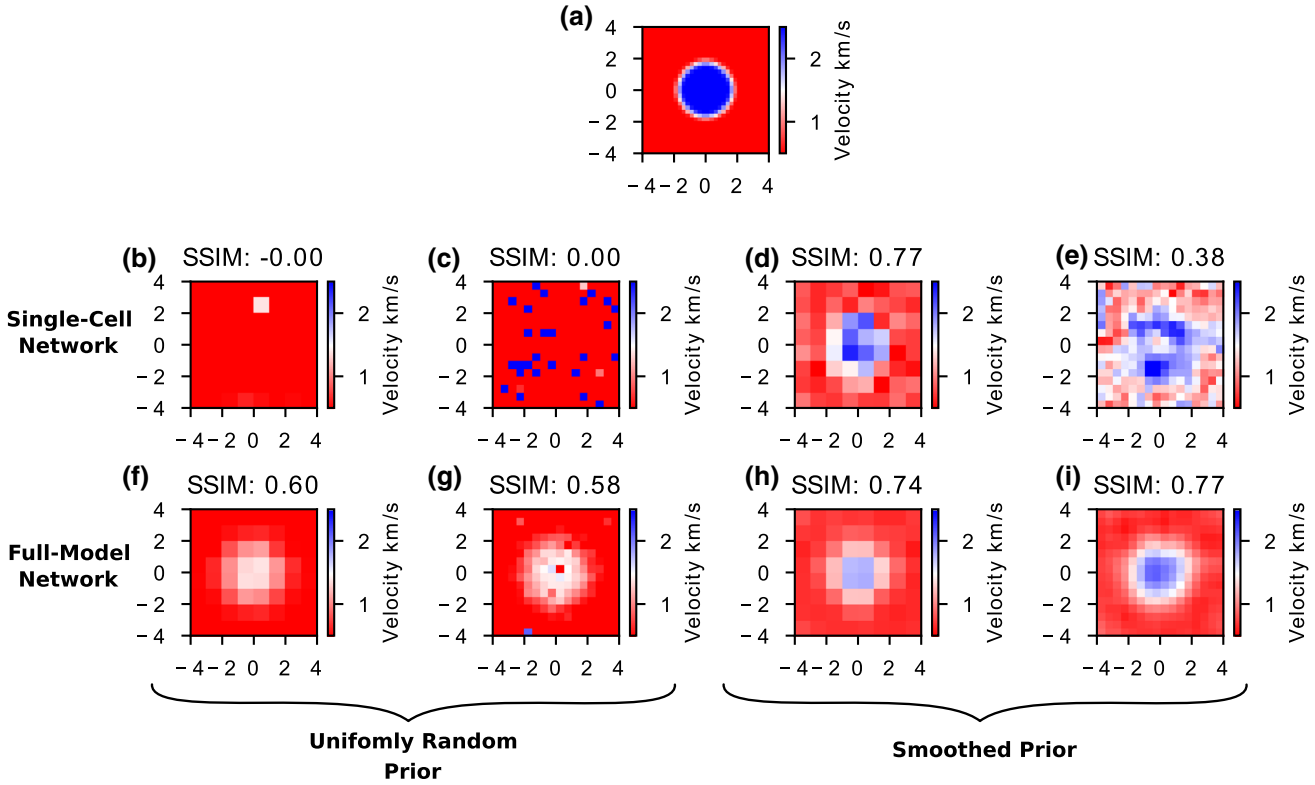

Fig. 14 **a** True velocity model. For a single-cell MDN, using a training set from a uniformly random distribution, results shown are mean velocities for **b** an 8×8 model and **c** a 16×16 model. Using the same network with a training set of spatially smoothed velocities, we obtain mean velocities for **d** an 8×8 model and **e** a 16×16 model. For a full-model MDN, using a training set from a uniformly random distribution, we obtain mean velocities for an **f** 8×8 model and **g** a 16×16 model. Using the same network with a training set of spatially smoothed velocities, we obtain mean velocities for an **h** 8×8 model and **i** a 16×16 model. Corresponding SSIM is shown above each result. The colour axis has been clipped to the velocity bounds of the training set (0.5 km/s, 2.5 km/s) (colour figure online)

velocity estimates and uncertainties as such information effectively reduces the dimensionality of the tomography problem. However, as with all Bayesian inversions if we impose false prior information, we can lose important information about uncertainties. By training networks to invert for a full tomographic model at once, we can also understand the relationship between velocities in neighbouring pixels; however, the number of parameters in the inversion increases substantially, and training for accurate models proves to be significantly more difficult. We compare the speed of neural network inversion to more standard Monte Carlo methods and determine that for many repeated inversions which occur in monitoring situations, MDNs may outperform Monte Carlo methods in terms of computational cost.

Acknowledgements The authors thank the Edinburgh Interferometry Project sponsors (Equinor, Schlumberger Cambridge Research and Total) for supporting this research.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix 1: Network configurations

The networks trained on individual cells used four fully connected layers (FC), where each node receives an input from every node in the previous layer. In between each node of the fully connected (FC) layers, a rectified linear

Table 1 Configurations of the networks with four fully connected (FC) layers

Size of model	FC 1	FC 2	FC 3	FC 4	Total no. of parameters
8×8	270	1000	380	600	1,544,765
	100	500	450	550	1,622,685
	800	325	100	300	1,165,660
	200	400	200	50	334,335
	300	250	200	50	331,685
	900	700	70	550	2,077,505
	200	250	200	50	274,185
	300	400	200	50	406,835
16×16	375	500	300	600	5,265,470
	300	250	200	50	625,445
	200	400	200	50	628,095
	800	1000	500	550	6,076,995

Each row in the table represents a separate network trained. Eight networks were trained for the 8×8 models and four networks for the 16×16 models

Table 2 Configurations of the convolutional networks with three convolutional (Conv) layers and four fully connected (FC) layers

Conv 1		Conv 2		Conv 3		FC 1	FC 2	FC 3	FC 4	Total no. of parameters	
Filter	Kernel	Filter	Kernel	Filter	Kernel					8×8	16×16
128	5	128	5	64	1	800	150	600	1500	4,717,405	13,363,165
32	9	32	5	16	1	500	300	600	1500	4,354,183	12,999,943
32	9	32	5	16	1	500	200	2000	1250	5,641,438	12,847,243
32	9	8	5	16	1	500	300	600	1750	4,986,575	15,054,335
32	9	32	5	16	1	500	1500	50	1250	3,528,333	10,734,093

Each row in the table represents a separate network trained

unit (ReLU) is used. The individual layer sizes and the total number of parameters to be trained in each networks are outlined in Table 1.

Networks trained on the whole model (all cells at once) used a convolutional network with three convolutional layers (Conv) and four fully connected layers. The sizes of each layer and the total number of parameters to be trained in each networks are outlined in Table 2.

Appendix 2: Structural similarity index measure (SSIM)

We use the form of the SSIM metric described in [45]. Let x and y be a window of $N \times N$ size. We calculate the luminance $l(x, y)$, contrast $c(x, y)$ and structure $s(x, y)$ defined as:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{7}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{8}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{9}$$

where μ and σ are the mean and variance of the windows x or y and σ_{xy} is the covariance of x and y. To avoid instability in the division, constants C_1, C_2 and C_3 are defined as $C_1 = (k_1 L)^2$ and $C_2 = (k_2 L)^2$ where L is the dynamic range of the cell values, while $k_1 = 0.01$ and $k_2 = 0.03$, and $C_3 = C_2/2$. The three components are combined to give the full SSIM:

$$\mathrm{SSIM}(x,y) = [l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma] \tag{10}$$

where α, β and γ are weighting parameters. Setting $\alpha = \beta = \gamma = 1$, we can simplify the expression to:

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{11}$$

We perform the calculation over sliding windows and take the mean of the resulting SSIM (x, y) values. For the 8×8 models, we use 3×3 windows and the 16×16 models use 7×7 windows, so that the windows cover a similar spatial area.

References

1. Aki K, Christoffersson A, Husebye ES (1977) Determination of the three-dimensional seismic structure of the lithosphere. J Geophys Res 82(2):277–296
2. Dziewonski AM, Woodhouse JH (1987) Global images of the earth's interior. Science 236(4797):37–48
3. Mordret A, Shapiro NM, Singh SS, Roux P, Barkved OI (2013) Helmholtz tomography of ambient noise surface wave data to estimate Scholte wave phase velocity at Valhall life of the field. Geophysics 78(2):WA99–WA109
4. Rawlinson N, Pozgay S, Fishwick S (2010) Seismic tomography: a window into deep earth. Phys Earth Planet Inter 178(3–4):101–135
5. Galetti E, Curtis A, Meles GA, Baptie B (2015) Uncertainty loops in travel-time tomography from nonlinear wave physics. Phys Rev Lett 114(14):148501
6. Bodin T, Sambridge M (2009) Seismic tomography with the reversible jump algorithm. Geophys J Int 178(3):1411–1436
7. Hawkins R, Sambridge M (2015) Geophysical imaging using trans-dimensional trees. Geophys J Int 203(2):972–1000
8. Piana Agostinetti N, Giacomuzzi G, Malinverno A (2015) Local three-dimensional earthquake tomography by trans-dimensional Monte Carlo sampling. Geophys J Int 201(3):1598–1617
9. Zhang X, Curtis A, Galetti E, de Ridder S (2018) 3-d Monte Carlo surface wave tomography. Geophys J Int 215(3):1644–1658
10. Zhang X, Hansteen F, Curtis A, de Ridder S (2019) 1-D, 2-D, and 3-D ambient noise tomography using a dense passive seismic array installed on the North Sea seabed. J Geophys Res 125:e2019JB018552
11. Devilee RJR, Curtis A, Roy-Chowdhury K (1999) An efficient, probabilistic neural network approach to solving inverse problems: inverting surface wave velocities for Eurasian crustal thickness. J Geophys Res Solid Earth 104(B12):28841–28857. https://doi.org/10.1029/1999JB900273
12. Käufl P, Valentine AP, de Wit RW, Trampert J (2016) Solving probabilistic inverse problems rapidly with prior samples. Geophys J Int 205(3):1710–1728. https://doi.org/10.1093/gji/ggw108
13. Sambridge M (1999) Geophysical inversion with a neighbourhood algorithm—II. Appraising the ensemble. Geophys J Int 138(3):727–746
14. Roth G, Tarantola A (1994) Neural networks and inversion of seismic data. J Geophys Res 99(B4):6753–6768
15. Bishop CM (1995) Neural networks for pattern recognition. Oxford University Press, Oxford
16. Moya A, Irikura K (2010) Inversion of a velocity model using artificial neural networks. Comput Geosci 36(12):1474–1483
17. Araya-Polo M, Jennings J, Adler A, Dahlke T (2018) Deep-learning tomography. Lead Edge 37(1):58–66. https://doi.org/10.1190/tle37010058.1
18. Gupta S, Kothari K, de Hoop MV, Dokmanić I (2018) Deep mesh projectors for inverse problems. arXiv preprint arXiv:180511718
19. Mairal J, Bach F, Ponce J et al (2014) Sparse modeling for image and vision processing. Found Trends® Comput Graph Vis 8(2–3):85–283
20. Bianco MJ, Gerstoft P (2018) Travel time tomography with adaptive dictionaries. IEEE Trans Comput Imaging 4(4):499–511
21. Kong Q, Trugman DT, Ross ZE, Bianco MJ, Meade BJ, Gerstoft P (2018) Machine learning in seismology: turning data into insights. Seismol Res Lett 90(1):3–14
22. Meier U, Curtis A, Trampert J (2007) Fully nonlinear inversion of fundamental mode surface waves for a global crustal model. Geophys Res Lett. https://doi.org/10.1029/2007GL030989
23. Meier U, Curtis A, Trampert J (2007) Global crustal thickness from neural network inversion of surface wave data. Geophys J Int 169(2):706–722. https://doi.org/10.1111/j.1365-246X.2007.03373.x
24. Meier U, Trampert J, Curtis A (2009) Global variations of temperature and water content in the mantle transition zone from higher mode surface waves. Earth Planet Sci Lett 282(1):91–101. https://doi.org/10.1016/j.epsl.2009.03.004
25. Shahraeeni MS, Curtis A (2011) Fast probabilistic nonlinear petrophysical inversion. Geophysics 76(2):E45–E58
26. Shahraeeni MS, Curtis A, Chao G (2012) Fast probabilistic petrophysical mapping of reservoirs from 3d seismic data. Geophysics 77(3):O1–O19
27. Käufl P, Valentine AP, O'Toole TB, Trampert J (2014) A framework for fast probabilistic centroid-moment-tensor determination–inversion of regional static displacement measurements. Geophys J Int 196(3):1676–1693
28. Käufl P, Valentine A, de Wit R, Trampert J (2015) Robust and fast probabilistic source parameter estimation from near-field displacement waveforms using pattern recognition. Bull Seismol Soc Am 105(4):2299–2312
29. De Wit RW, Valentine AP, Trampert J (2013) Bayesian inference of earth's radial seismic structure from body-wave traveltimes using neural networks. Geophys J Int 195(1):408–422. https://doi.org/10.1093/gji/ggt220
30. Nawaz MA, Curtis A (2017) Bayesian inversion of seismic attributes for geological facies using a hidden Markov model. Geophys J Int 208(2):1184–1200
31. Nawaz MA, Curtis A (2018) Variational Bayesian inversion (VBI) of quasi-localized seismic attributes for the spatial distribution of geological facies. Geophys J Int 214(2):845–875
32. Nawaz M, Curtis A (2019) Rapid discriminative variational Bayesian inversion of geophysical data for the spatial distribution of geological properties. J Geophys Res Solid Earth 124(6):5867–5887
33. Cao R, Earp S, de Ridder SA, Curtis A, Galetti E (2020) Near-real-time near-surface 3d seismic velocity and uncertainty models by wavefield gradiometry and neural network inversion of ambient seismic noise. Geophysics 85(1):KS13–KS27
34. Tarantola A (2005) Inverse problem theory. SIAM, Philadelphia
35. Dietterich TG (2000) Ensemble methods in machine learning. In: International workshop on multiple classifier systems. Springer, pp 1–15
36. Rawlinson N, Sambridge M (2004) Wave front evolution in strongly heterogeneous layered media using the fast marching method. Geophys J Int 156(3):631–647
37. Rawlinson N, Sambridge M (2005) The fast marching method: an effective tool for tomographic imaging and tracking multiple phases in complex layered media. Explor Geophys 36(4):341–350
38. Curtis A, Lomax A (2001) Prior information, sampling distributions, and the curse of dimensionality. Geophysics 66(2):372–378
39. Bergstra J, Komer B, Eliasmith C, Yamins D, Cox DD (2015) Hyperopt: a python library for model selection and hyperparameter optimization. Comput Sci Discov 8(1):014008
40. Curtis A, Wood R (2004) Geological prior information: informing science and engineering. Geological Society of London, London
41. Walker M, Curtis A (2014) Expert elicitation of geological spatial statistics using genetic algorithms. Geophys J Int 198:342–356
42. Mosser L, Dubrule O, Blunt MJ (2018) Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. arXiv p arXiv:1806.03720

43. Walker M, Curtis A (2014) Varying prior information in Bayesian inversion. Inverse Probl 30(6):065002
44. Galetti E, Curtis A, Baptie B, Jenkins D, Nicolson H (2017) Transdimensional love-wave tomography of the British Isles and shear-velocity structure of the East Irish Sea Basin from ambient-noise interferometry. Geophys J Int 208(1):36–58
45. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP et al (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612