Geophysical Journal International

Geophys. J. Int. (2024) **236,** 565–592
Advance Access publication 2023 October 30
GJI Applied and Marine Geophysics

https://doi.org/10.1093/gji/ggad424

Iterative multitask learning and inference from seismic images

Kai Gao

Earth and Environmental Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545*, USA. E-mail: kaigao@lanl.gov*

Accepted 2023 October 24. Received 2023 January 24; in original form 2023 August 1

SUMMARY

Seismic interpretation aims to extract quantitative and interpretable attributes from a seismic image produced using some migration method to inform characteristics of a subsurface reservoir or target of interest. Current paradigms for computing seismic attributes mostly rely on single-task algorithms. We develop an iterative, multitask machine learning method to learn and infer multiple attributes from a seismic image. This method is composed of two stages: a multitask inference stage and a multimodal, multitask refinement stage. The basic mechanism of this method is that we train a multitask inference neural network to estimate a set of attributes, including a relative geological time volume, a denoised higher-resolution seismic image and multiple fault attributes (including probability, dip and strike), from a low-resolution, noisy seismic image; then we input the inferred attributes to a multitask refinement NN to enhance the raw inference results iteratively. The two multitask neural networks are trained separately based on synthetic seismic images and associated labels generated by geological modelling. Applications of this multitask learning and inference method to synthetic and field seismic images show that our method can improve the structural consistency among output seismic attributes compared with single-task neural networks, leading to more reliable automatic interpretation and subsurface characterization.

Key words: Multitask machine learning; Automatic seismic interpretation; Relative geological time; High-resolution image; Denoising; Geological fault characterization.

1 INTRODUCTION

Initially thrived in computer vision and natural language processing (e.g. LeCun *et al.* 1998; Goodfellow *et al.* 2014; Lai *et al.* 2015; Krizhevsky *et al.* 2017), machine learning (ML) has introduced many new insights and methods to earthquake seismology (e.g. Zhu & Beroza 2018; Ross *et al.* 2018; Mousavi *et al.* 2020), seismogenesis and induced seismicity (e.g. Rouet-Leduc *et al.* 2019; Johnson *et al.* 2021; Qin *et al.* 2022), seismic interpretation (e.g. Wu *et al.* 2019a; Di *et al.* 2022), sophisticated geophysical modelling (e.g. Song *et al.* 2021; bin Waheed *et al.* 2021) and subsurface medium property characterization (e.g. Wu & Lin 2020; Sun & Alkhalifah 2020; Leong & Zhu 2021), to name a few.

Quantitative seismic interpretation aims to extract quantitative seismic attributes from seismic images to build geologically interpretable models to facilitate subsurface characterization. Seismic interpretation is especially important to subsurface energy exploration, where accurate location and characterization of the geology of interest is crucial to the success of field operations. While mostly relying on hand picking and qualitative inference in the early days, seismic interpretation now enjoys significantly improved accuracy and automation thanks to the development of various dedicated and efficient algorithms, and more recently, machine learning. In this work, we focus on extracting the following seismic attributes from a single seismic image using machine learning: a relative geological time (RGT) volume, a denoised higher-resolution (DHR) image and multiple fault attributes (probability, dip and strike).

RGT refers to an attribute extracted from a seismic image volume to indicate the relative stratigraphic time of reflectors and structures (Stark 2004; Bi *et al.* 2021). A larger RGT value indicates older stratigraphic time while a smaller RGT value indicates newer stratigraphic time. Because RGT is a direct representation of the stratigraphic order in a region, isosurfaces extracted from an RGT volume should be consistent with reflectors and horizons in a seismic image. Such chronological information is important for such as seismic horizon flattening (Lomask *et al.* 2006), stratigraphic analysis (Lou *et al.* 2020) and seismic interpretation for complex geological structures (Di *et al.* 2022). Importantly, RGT can indicate the spatial location and displacement of faults or to indicate unconformities by finding lateral discontinuities, which is important for building an accurate geological model. However, to build a complete RGT volume, sophisticated post-processing and merging algorithms are usually needed following the computation of RGT based on the instantaneous phase of each trace. Bi *et al.* (2021) use a fully convolutional neural network (NN) trained by a structural similarity loss function (Wang *et al.* 2004) to estimate RGT directly from a seismic image. Their work is based on

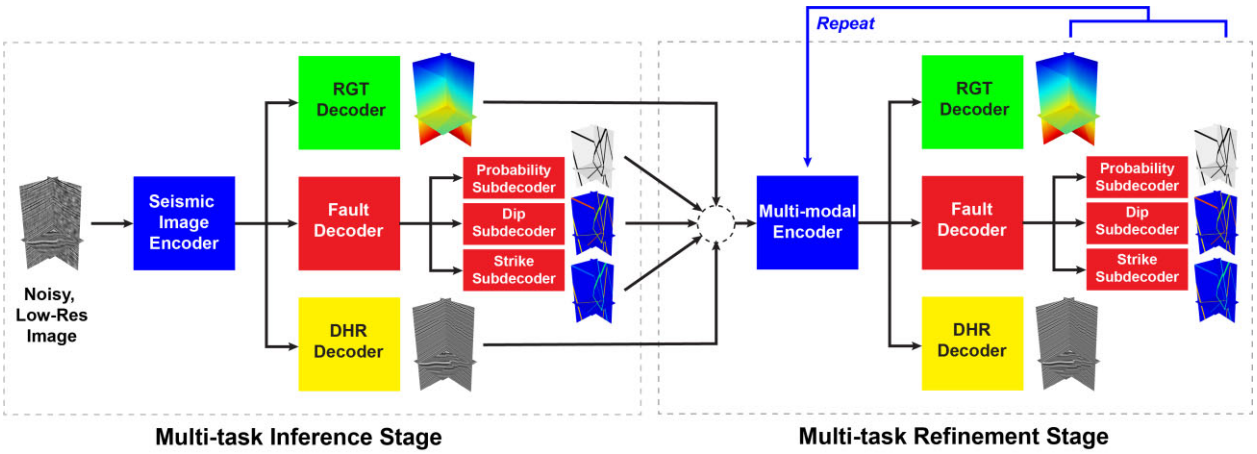

Figure 1. Schematics of the multitask inference net (MTI-Net) and the multitask refinement net (MTR-Net). RGT refers to relative geological time, while DHR refers to denoised higher-resolution image. Fault strike inference and refinement only apply to 3-D images. Dashed circle in the centre means concatenation. See Fig. 2 for details. 'Repeat' means multiple subsurface attributes generated by MTR-Net are fed to MTR-Net for further refinement.

the traditional U-Net, and uses the so-called attention mechanism to improve the learning and generalization capability for complex images.

Denoising post-stack seismic images has been a long-standing problem in the seismic processing workflow. Relevant methods consist of a large reservoir that we are unable to fully review in this work, and we only review some of the works that are relevant to our work and are machine-learning-based. Kaur *et al.* (2020) developed a CycleGAN-based seismic denoiser to approximate seismic denoising as a generative modelling processing from a noisy image to a noise-free image. CycleGAN (Zhu *et al.* 2017) is an evolved version of the generative adversarial network (GAN; Goodfellow *et al.* 2014). Although GAN and CycleGAN are powerful generative neural networks with remarkable denoising capability, they are usually expensive and sometimes even unstable to train because they are two neural network components (generator and discriminator) involved in the process. Gao *et al.* (2022a) demonstrated that it is feasible to use a supervised autoencoder-decoder NN with a nested U-Net structure to perform image denoising and resolution enhancement simultaneously, and thus improve the convergence of a least-squares reverse-time migration.

Geological faults are crucial for oil and gas exploration (e.g. Di & Gao 2017), geothermal system exploration (e.g. Gao *et al.* 2021), geological and geodynamics studies (e.g. Cheng *et al.* 2021) and so on. Yet fault characterization is one of the most challenging tasks in automatic seismic interpretation. Traditional methods or seismic attributes for identifying geological faults include semblance (Marfurt *et al.* 1998), coherence (Marfurt *et al.* 1999), local fault extraction (Cohen *et al.* 2006), ant tracking (Pedersen *et al.* 2005), structure tensor (Wu 2017), discontinuity-preserved anisotropic diffusion filtering (Wu & Guo 2018) and optimal surface voting (Wu & Fomel 2018), to name a few. These approaches rely on different underlying principles and algorithms and differ in the accuracy. With the boost of machine learning, fault identification enters a new phase, where neural networks that were developed for computer vision tasks have started to automate fault identification. Early works on applying machine learning to fault detection (e.g. Xiong *et al.* 2018; Di *et al.* 2018) focus on patch-by-patch classification model for an input seismic image. Wu *et al.* (2019a) developed a convolutional neural network based on U-Net (Ronneberger *et al.* 2015) to automatically extract fault probability from a seismic image in an end-to-end fashion. Their so-called FaultSeg3D NN treats fault detection as a binary semantic segmentation problem, resulting in a significant computation efficiency and accuracy promotion compared with earlier approaches. Numerous subsequent works improve FaultSeg3D, most of which are based on more sophisticated architectures (e.g. Gao *et al.* 2022c, d) or improved data preparation and transfer learning strategies (Li *et al.* 2019; Cunha *et al.* 2020; Zhang *et al.* 2022). Wang *et al.* (2022) developed a knowledge distillation strategy along with a synthetic-field seismic image mixing strategy to improve the performance of convolutional neural network (CNN) based fault detection. These machine-learning-based methods, however, mainly focus on the detection of faults through image segmentation rather than characterizing the geometrical attributes (e.g. dip, strike) of the faults, independently or simultaneously.

Despite the powerful and accurate inference capability offered by these existing machine learning models, they are mostly single-task, meaning that the NNs associated with these tasks usually perform a single functionality at a time based on an input seismic image. To estimate multiple attributes, one must train multiple independent NNs and perform independent inferences using each of them. It is generally difficult to ensure structure consistency among produced results by estimating these attributes independently with single-task NNs. A natural and intriguing question is whether it is possible to train a multitask NN to perform all the tasks simultaneously. Another important question is whether such a multitask NN will lead to more structurally consistent estimation of different attributes by explicit or implicit constraints.

Multitask learning (MTL) is a notable machine-learning paradigm that learns multiple objectives from a common representation of the input (Kendall *et al.* 2018; Vandenhende *et al.* 2022). MTL can be viewed as a form of inductive transfer that introduces inductive bias to an NN—the learning for a specific task in an MTL can be implicitly constrained by other learning tasks, leading to improved consistency among different outputs. There has been only limited research on developing a multitask NN to perform end-to-end seismic image enhancement and seismic interpretation. Wu *et al.* (2019c) developed a CNN to simultaneously output the fault attributes (probability, strike and dip) from a seismic image. However, their work is based on a regression CNN consisting of both convolutional layers and fully connected layers.

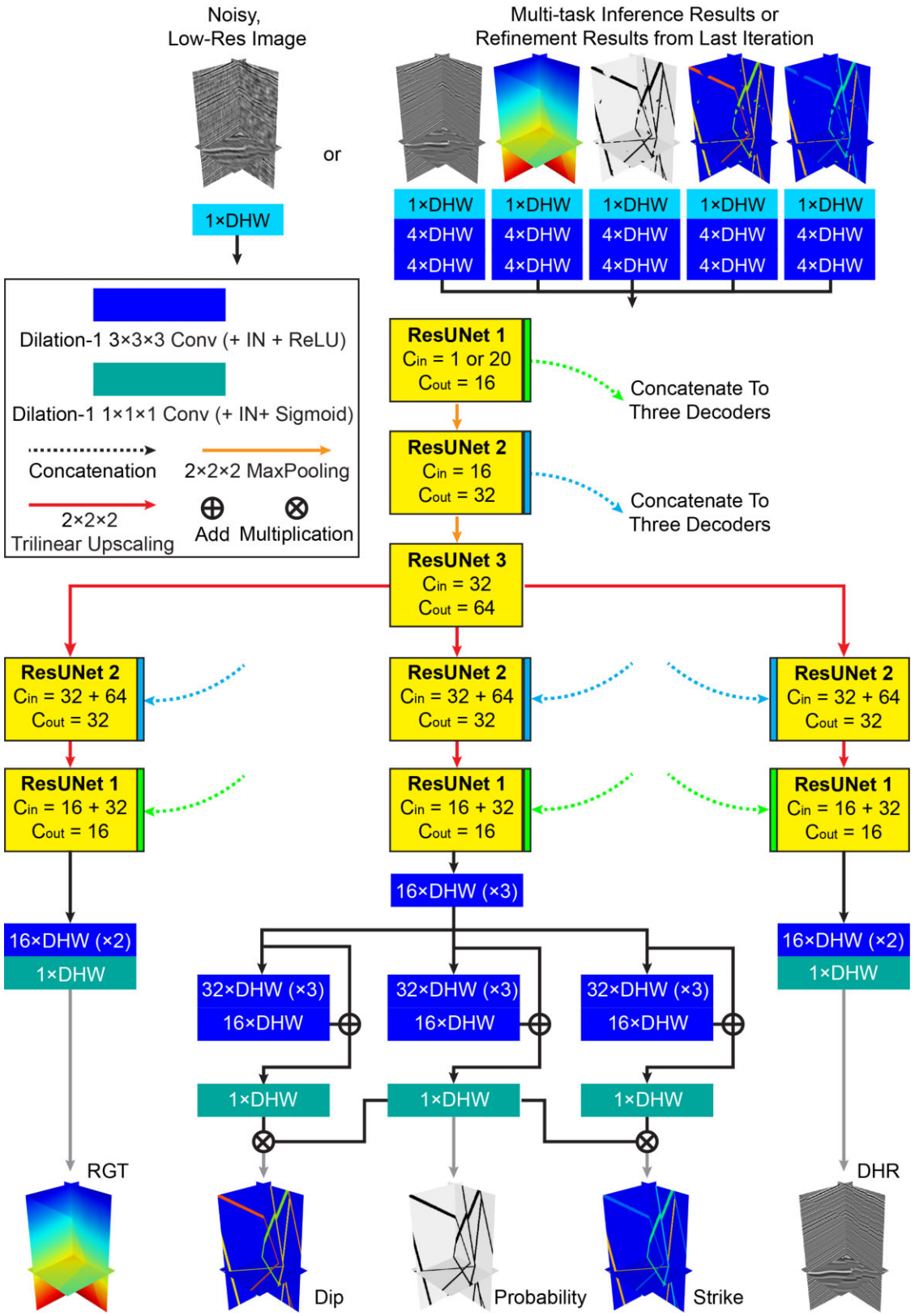

Figure 2. Architecture of MTI-Net and MTR-Net in Fig. 1. MTI-Net and MTR-Net share the same encoder-decoder structure but differ in the input. The input to MTI-Net is a single seismic image, while the inputs to the MTR-Net include a seismic image, a RGT volume and fault attributes. '$C \times DHW/n$' means a convolutional layer with C channels and a depth, height, width of D/n, H/n, W/n, respectively. The dimensions D, H and W can be any positive integer values. The final convolutional layers (green blocks) for dip and strike are followed by instance normalization (IN); in addition, there is a sigmoid activation layer following the final convolutional layer or IN for generating RGT, fault dip, probability, and strike. Architectures of the residual U-Nets, namely ResUNet 1, 2 and 3, are displayed in Fig. A1.

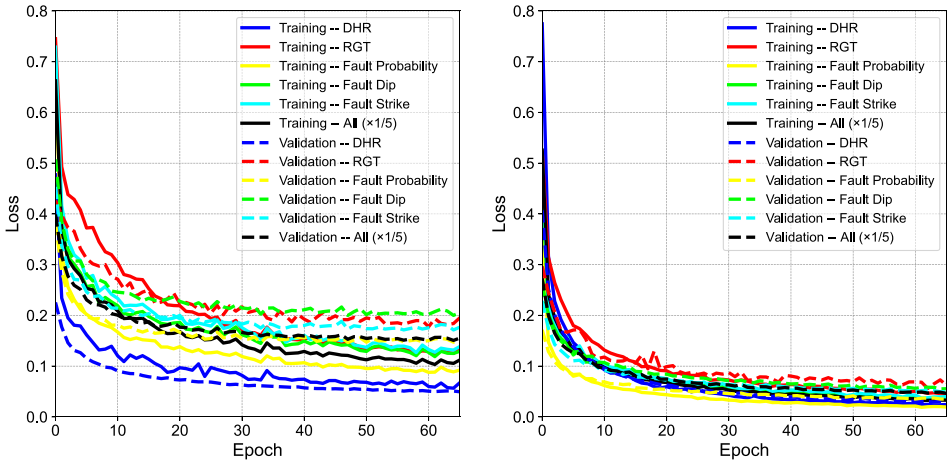

Figure 3. Losses associated with the training and validation of 3-D (a) MTI-Net and (b) MTR-Net. The total losses in both panels are scaled by 1/5 for better visualization.

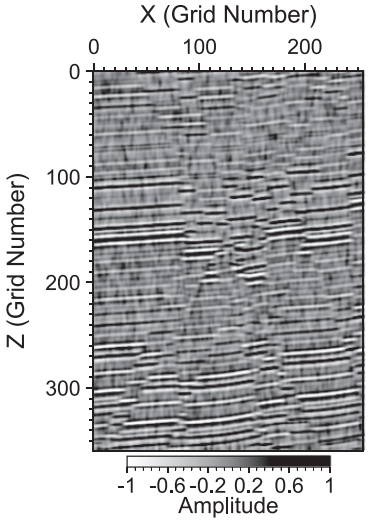

Figure 4. A 2-D noisy, low-resolution image in the validation set with a size of $N_z \times N_x = 360 \times 256$ gridpoints.

In addition to higher training costs caused by the fully connected layers, the computation must be performed patch by patch during the inference phase, leading to high computational costs. Li *et al.* (2022) developed a U-Net-based fully convolutional NN to perform simultaneous image denoise and image resolution enhancement. Nevertheless, the work focuses on 2-D image denoising and resolution enhancement. Wu *et al.* (2019b) developed a multitask NN to estimate fault probability, denoise the input image and estimate the normal of reflectors. The NN does not, however, enhance the resolution of the input image. To our knowledge, there has been no end-to-end, fully convolutional NN that can achieve all the tasks (image denoising, image resolution enhancement, RGT estimation and multiple fault attributes estimation) simultaneously for a 3-D seismic image.

In this work, we achieve multitask learning and inference through a novel iterative multitask ML method, which can simultaneously denoise an input seismic image, enhance the resolution of the image, infer a RGT and multiple fault attributes from this image. Our iterative multitask ML method consists of two stages: a multitask inference stage and a multimodal, multitask refinement stage. In multitask inference, we develop and train a supervised fully convolutional NN consisting of one encoder module and multiple parallel decoder modules to infer RGT, DHR and fault attributes from a low-resolution, noisy seismic image. We call the NN associated with this stage a multitask inference NN, or MTI-Net for short. A notable issue with ML-based automatic seismic interpretation is that the estimated attributes may contain 'noise'. For instance, ML-based fault detection methods (Wu *et al.* 2019a; Gao *et al.* 2022d, c) may generate false fault detection results that appear as randomly scattered, small-scale faults that are in general difficult to interpret. Meanwhile, there may be numerous discontinuities along a detected fault path or fault surface, leading to 'cheese-hole' artefacts in faults which are not particularly geological plausible—in realistic geology, a fault was formed by relative displacement of geological blocks due to tectonic stress distribution and therefore should more of less form a smoothly varying surface or path. However, because most of supervised ML-based interpretation methods work in a single-pass manner (e.g. one inputs an seismic image and the NN outputs some attributes), there is no systematic approach to guarantee the outputs from an NN free of such artefacts and noise. For this reason, in the multitask refinement stage, we develop an innovative multimodal, multitask NN, or MTR-Net for short, to iteratively refine the MTI-Net inference results. The inputs to MTR-Net in the first iteration include the DHR image, the RGT volume and fault attributes generated by the MTI-Net (i.e. MTR-Net is multimodal) and the outputs from MTR-Net include the refined attributes (i.e. MTR-Net is multitask). Subsequent iterations refine the outputs from MTR-Net itself, thus resulting in gradually enhanced multitask interpretation results.

In terms of architectures of MTI-Net and MTR-Net, inspired by previous works in natural image segmentation (Qin *et al.* 2020) and fault detection (Gao *et al.* 2022c), we design a nested residual U-Net structure to achieve a large receptive field for both MTI-Net and MTR-Net. For fault characterization, our MTI/MTR-Net can infer pixel-wise geometrical properties (dip, strike) associated with the faults in addition to inferring pixel-wise fault probability, providing a novel fully convolutional approach for accurate end-to-end geological fault system characterization. The pixel-wise geometrical property inference is in contrast to previous works that estimate the fault attributes patch by patch (Wu *et al.* 2019c). Regarding

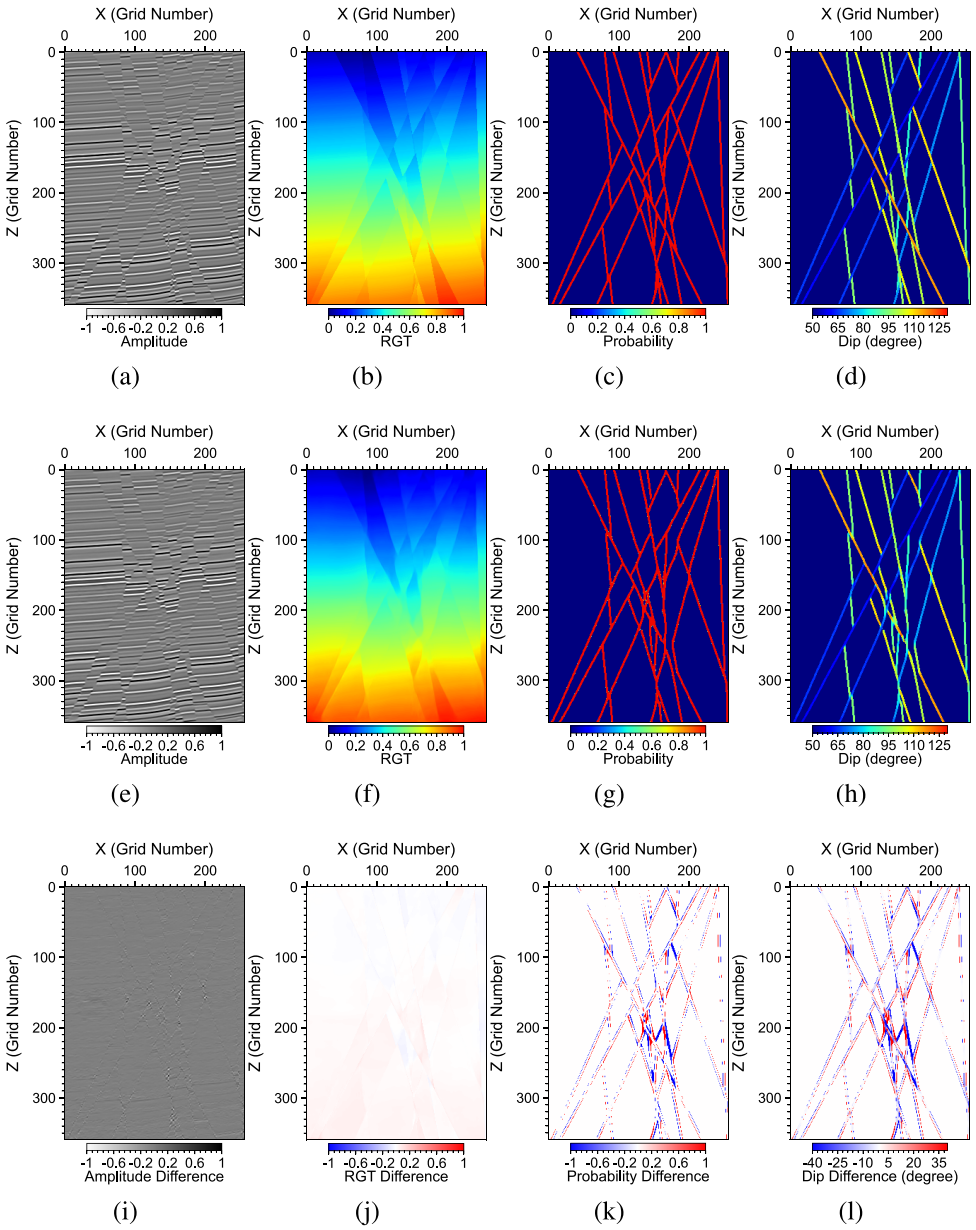

Figure 5. Demonstration of 2-D MTI-Net applied to the example image displayed in Fig. 4. Panels (a-d) display the labels. Panels (e-h) display the inference results produced by MTI-Net. Panels (i–l) display the differences between the MTI-Net results and labels.

training MTI/MTR-Net, we use a hybrid loss function to simultaneously learn different labels where different labels are assigned with different weights. We demonstrate the efficacy and accuracy of our new multitask learning method using synthetic and field data images.

The rest of the paper is organized as follows. In Section 2, we describe the architecture of MTI-Net and MTR-Net and describe the loss functions and algorithms we used for training these two NNs. In Section 3, we use synthetic seismic images and images generated from field seismic data to validate the efficacy and accuracy of our MTI-Net and MTR-Net trained with synthetic seismic images and labels and discuss several potential issues in the training and application of our multitask ML method to different data. We then conclude the paper by summarizing our findings in Section 4.

2 METHODOLOGY

2.1 Architecture

Our multitask inference-refinement neural network consists of two parts: a multitask inference net (MTI-Net), and a multitask refinement net (MTR-Net), as displayed in Fig. 1. MTI-Net is a multitask neural network with a single input (a 2-D or 3-D seismic image) and multiple outputs, including a relative geological time (RGT) volume, a denoised higher-resolution (DHR) image and three fault attributes (probability, dip and strike); fault strike is only estimated in the 3-D case. All the outputs have the same spatial dimensions (i.e. $N_z \times N_y \times N_x$) as the input seismic image. MTR-Net is a multimodal, multitask NN with multiple inputs, including a seismic image, a RGT volume and three fault attributes and with multiple outputs. Fig. 1 displays a schematic of the overall structure of this

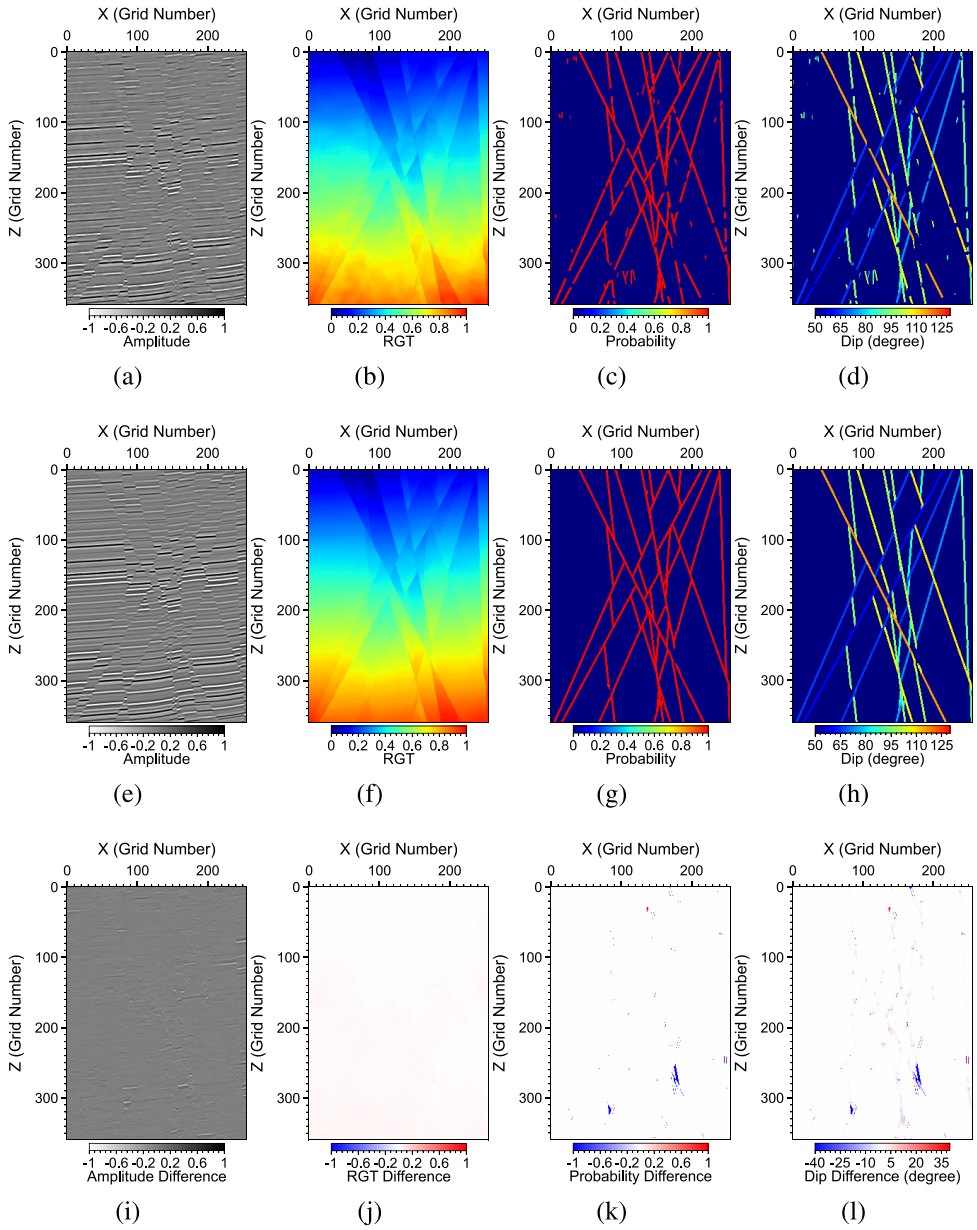

Figure 6. (a–d) Inputs to and (e–h) outputs from MTR-Net corresponding to the example image in Fig. 4. The labels are displayed in Figs 5(a)–(d). Panels (i–l) display the differences between the MTR-Net results and labels.

multitask inference-refinement neural network. The basic workflow of this multitask inference-refinement NN is that we first infer multiple attributes (RGT, DHR and fault attributes) from a seismic image using MTI-Net, then refines these outputs using MTR-Net iteratively. The second step (i.e. multitask refinement) can repeat several times to obtain gradually improved results by 'healing' the holes of estimated faults and removing scattered, small faults that are not correlated with lateral discontinuities of the image. The iterations can stop when the outputs from MTR-Net are close enough to those of the last iteration. We also intend to improve the structural consistency among RGT, DHR and fault attributes by using MTR-Net.

Both MTI-Net and MTR-Net are supervised fully convolutional encoder-decoder NNs (Murphy 2022) where we use a single autoencoder module to learn the features of the input (or inputs) and use multiple independent decoder modules to produce multiple attributes. Although there are many capable encoder-decoder structures, we use a nested structure to achieve encoding and decoding. In this nested NN, the encoder contains three spatial levels and the feature map encoder at each spatial level is a simplified U-Net with residual connection (He *et al.* 2016), which indicates that MTI/MTR-Net is a U-Net of U-Nets that has multiple decoder branches. The structure is inspired by pioneering works in natural image segmentation and seismic fault detection (Qin *et al.* 2020; Gao *et al.* 2022c), and has been demonstrated to be superior in learning complex features compared with the conventional U-Net architecture (Ronneberger *et al.* 2015).

Fig. 2 displays the detailed architectures of MTI/MTR-Net. The encoder module contains three residual U-Net blocks, where each residual U-Net block, or ResUNet for short, is a simplified U-Net

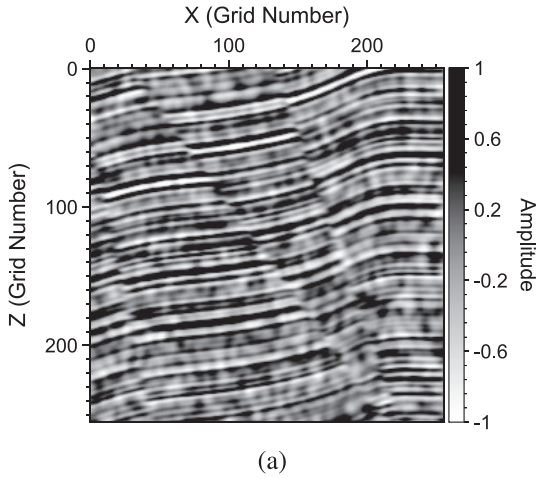

Figure 7. A 2-D noisy image with a size of $N_z \times N_x = 256 \times 256$ gridpoints. The image is not in the training or validation set.

with a residual connection. Residual connection is an effective approach to avoid the gradient vanishing/exploding issue (He *et al.* 2016), leading to improved performance of deep NN. For instance, ResUNet 1 is a residual U-Net consisting of four spatial levels as displayed in Fig. A1(a). Assuming the single-channel input to MTI-Net has a dimension of N along all spatial dimensions, for ResUNet 1 the lowest-level feature maps have a dimension of $N/2^3$ with 256 channels. ResUNet-2 and ResUNet-3 (displayed in Figs A1b and c, respectively) have a similar residual U-Net structure as that of ResUNet-1. However, for ResUNet-3, there are no downsampling or upsampling but only dilation convolutional layers with different dilation ratios. The purpose for designing such a dilation ResUNet block is to increase the receptive field. Meanwhile, we use instance normalization (Ulyanov *et al.* 2016) rather than batch normalization (Ioffe & Szegedy 2015) in our MTI/MTR-Net, which is more suitable for small-batch training. Mathematical details of these ResUNets can be found in Appendix A.

MTI/MTR-Net has three parallel decoder modules as displayed in the lower half of Fig. 2: a RGT decoder branch, a DHR decoder branch and a fault attributes decoder branch. Each module consists of ResUNet-1 and ResUNet-2 with structures displayed in Fig. A1. RGT and DHR decoders have almost identical architectures, and both output a single-channel attribute. Their only difference is that for the RGT decoder, we have a sigmoid activation layer at the end of decoder so that the output falls in the value range of [0, 1]. The fault decoder has a similar structure with that of the RGT and DHR, but we append three subdecoders following the fault decoder to generate three pixel-wise attributes: fault probability, dip and strike. To be specific, we append a residual convolutional block consisting of three $32 \times DHW$ (where 32 is the number of output channels, or filters) convolutional layers and one $16 \times DHW$ convolutional layer, followed by a $1 \times DHW$ convolutional layer with a kernel size of $1 \times 1 \times 1$ and a sigmoid activation layer. In addition, we multiply the fault dip and strike output from the sigmoid activation layers with the fault probability to enforce the consistency of spatial shape between the fault probability and the fault dip/strike. This NN structure indicates that during the training, fault probability acts like a regularizer on the fault dip/strike learning. For 2-D multitask fault attribute estimation, we only infer the fault probability and fault dip. Therefore, the 2-D version of MTI/MTR-Net only has two subdecoders following the fault decoder.

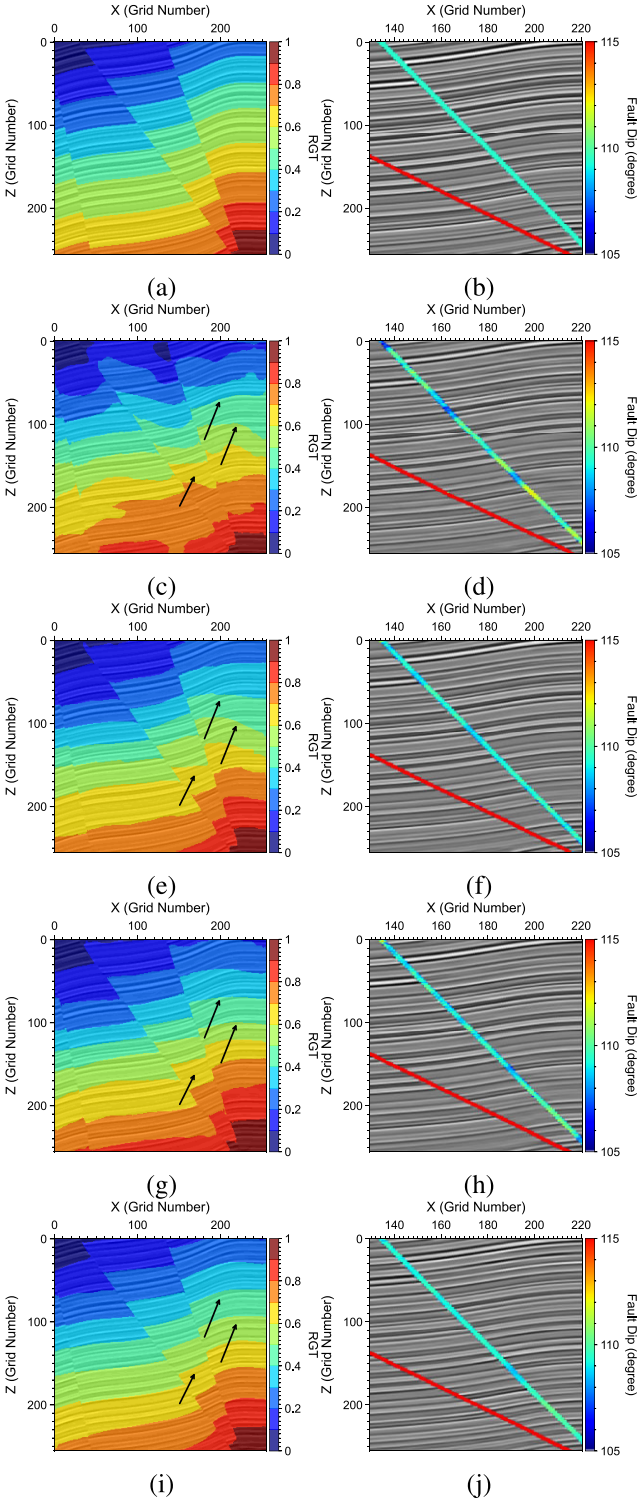

Figure 8. (a, b) The RGT and part of the fault dip labels associated with the image displayed in Fig. 7. Inferred RGT and fault dip by (c, d) U-MTI, (e, f) STI, (g, h) MTI-Net and (i, j) MTR-Net after three iterations with the MTI-Net results as the input.

In both 2-D and 3-D MTI/MTR-Nets, we build skip connections between the encoder branch to the three decoder branches via concatenation, as is done in the conventional U-Net. Thus, the structural consistency among the RGT, DHR and fault attributes are implicitly enforced during training and inference. The 2-D MTI-Net

Figure 9. A 3-D noisy, low-resolution image in the validation set.

contains approximately 4.8 million trainable parameters, while the 3-D MTI-Net contains approximately 14.5 million trainable parameters. Because MTR-Net only modifies the input part of MTI-Net with several additional convolutional blocks, it contains approximately the same number of trainable parameters with MTI-Net in both 2-D and 3-D.

To demonstrate MTI/MTR-Net, we also construct a conventional U-Net-based multitask inference NN, where we replace the encoder/decoder modules built upon nested residual U-Net blocks using a conventional U-Net that has a four-level structure (Wu *et al.* 2019a). At each level, the encoder is a simple convolutional block consisting of two convolutional layers followed by a ReLU activation and batch normalization. The encoder features maps are connected to three separate decoder branches via skip connection (i.e. concatenation). In addition, we need to compare the results generated using MTI-Net and single-task NNs. We implement and train three single-task NNs by simply turning off two of the decoder modules each time based on the same MTI-Net architecture display in Fig. 2. To simplify the terminologies, we call the multitask learning and inference NN that is built on the conventional U-Net U-MTI, and call the three single-task NNs that are built on nested residual U-Net blocks STI. We apply the same loss function and optimizer detailed below to all the four NNs, including U-MTI, STI, MTI-Net and MTR-Net.

2.2 Loss functions

Supervised learning needs a proper loss function to effectively and efficiently learn the mapping between the input and target (Murphy 2022). For multitask learning, different outputs have their unique characteristics and ranges of values. RGT is a mostly low-wavenumber field containing a limited number of high-wavenumber jumps in the horizontal directions. DHR image is oscillatory in the vertical direction and mostly smooth in the horizontal directions but may also contain discontinuities in the horizontal directions. Meanwhile, fault attributes are spatially sparse. We adopt a hybrid loss function to train MTI/MTR-Net:

$$\mathcal{L} = w_1\mathcal{L}_{\text{RGT}} + w_2\mathcal{L}_{\text{DHR}} + w_3\mathcal{L}_{\text{Fault Probability}} + w_4\mathcal{L}_{\text{Fault Dip}} + w_5\mathcal{L}_{\text{Fault Strike}}. \quad (1)$$

Studies show that for multitask learning, loss weighting may provide better training (e.g. Vandenhende *et al.* 2022). In this study, we choose $w_1 = 5$, $w_2 = w_4 = w_5 = 10$ and $w_3 = 1$ based on trial-and-error tests.

Specifically, to learn RGT, we use a hybrid loss function consisting of structural similarity index measure (SSIM; Wang *et al.* 2004) and L_1-norm losses:

$$\begin{aligned}\mathcal{L}_{\text{RGT}} &= 1 - \mathcal{S}(r, \hat{r}) + \mathcal{L}_1(r, \hat{r}),\\ &= 1 - \frac{(2\mu_r\mu_{\hat{r}} + c_1)(2\sigma_{r\hat{r}} + c_2)}{(\mu_r^2 + \mu_{\hat{r}}^2 + c_1)(\sigma_r^2 + \sigma_{\hat{r}}^2 + c_2)} + \frac{1}{N}\sum_{i=1}^{N}|r_i - \hat{r}_i|, \quad (2)\end{aligned}$$

where $\mathcal{S}$ stands for SSIM; μ_r and $\mu_{\hat{r}}$ are the means of the inferred RGT r and the ground truth RGT $\hat{r}$ volumes, respectively; σ_r and $\sigma_{\hat{r}}$ are the standard deviations of r and $\hat{r}$, respectively; $\sigma_{r\hat{r}}$ is the covariance of r and $\hat{r}$; $c_1 = (k_1R)^2$ and $c_2 = (k_2R)^2$ are two stabilization parameters to avoid division by zero with $k_1 = 0.01$, $k_2 = 0.03$ and $R = \hat{r}_{\max} - \hat{r}_{\min}$ is the dynamical range of the RGT volume. In the training, we prepare all RGT labels so that each RGT volume varies from 0 to 1, and therefore, $R = 1$. In eq. (2), $N = N_zN_yN_x$ is the total number of pixels of the input image (or equivalently, any of the five labels).

To learn DHR, we use an L_2-norm loss function:

$$\mathcal{L}_{\text{DHR}} = \mathcal{L}_2(u, \hat{u}) = \frac{1}{N}\sum_{i=1}^{N}(u_i - \hat{u}_i)^2, \quad (3)$$

where N is the total number of pixels in the inferred image u and the ground-truth image $\hat{u}$.

We consider estimating the fault probability as a binary semantic segmentation problem with one target class. Because the faults (fault pixels) and the background (non-fault pixels) are highly unbalanced in terms of pixel counts, we adopt a smooth dice loss function (Sudre *et al.* 2017; Jadon 2020) to learn fault probability:

$$\mathcal{L}_{\text{Fault Probability}} = 1 - \mathcal{D}(f, \hat{f}) = 1 - \frac{2\sum_{i=1}^{N} f_i\hat{f}_i + 1}{\sum_{i=1}^{N} f_i + \sum_{i=1}^{N}\hat{f}_i + 1}, \quad (4)$$

where $\mathcal{D}$ stands for the smooth dice and f and $\hat{f}$ represent the inferred and the ground-truth fault pixels, respectively. Meanwhile, to learn fault dip and fault strike, we use L_1-norm loss functions:

$$\mathcal{L}_{\text{Fault Dip}} = \mathcal{L}_1(h, \hat{h}) = \frac{1}{N}\sum_{i=1}^{N}|h_i - \hat{h}_i|, \quad (5)$$

$$\mathcal{L}_{\text{Fault Strike}} = \mathcal{L}_1(s, \hat{s}) = \frac{1}{N}\sum_{i=1}^{N}|s_i - \hat{s}_i|, \quad (6)$$

where h and $\hat{h}$ represent the inferred and the ground-truth fault dip, respectively; s and $\hat{s}$ represent the inferred and the ground-truth fault strike, respectively. Again, the fault dip and strike attributes are implicitly constrained by the fault probability attribute. Therefore, the losses defined in eqs (5) and (6) implicitly constrains the inference of the fault probability as well.

Alternative loss functions (and their combinations) different from what we use here may also be used to train MTI/MTR-Net, but it is out of the focus of this work to perform a thorough comparison of all possible loss functions and combinations. The loss functions we adopt here are chosen based on extensive trial-and-error tests and several previous studies (Bi *et al.* 2021; Gao *et al.* 2022a, b, c).

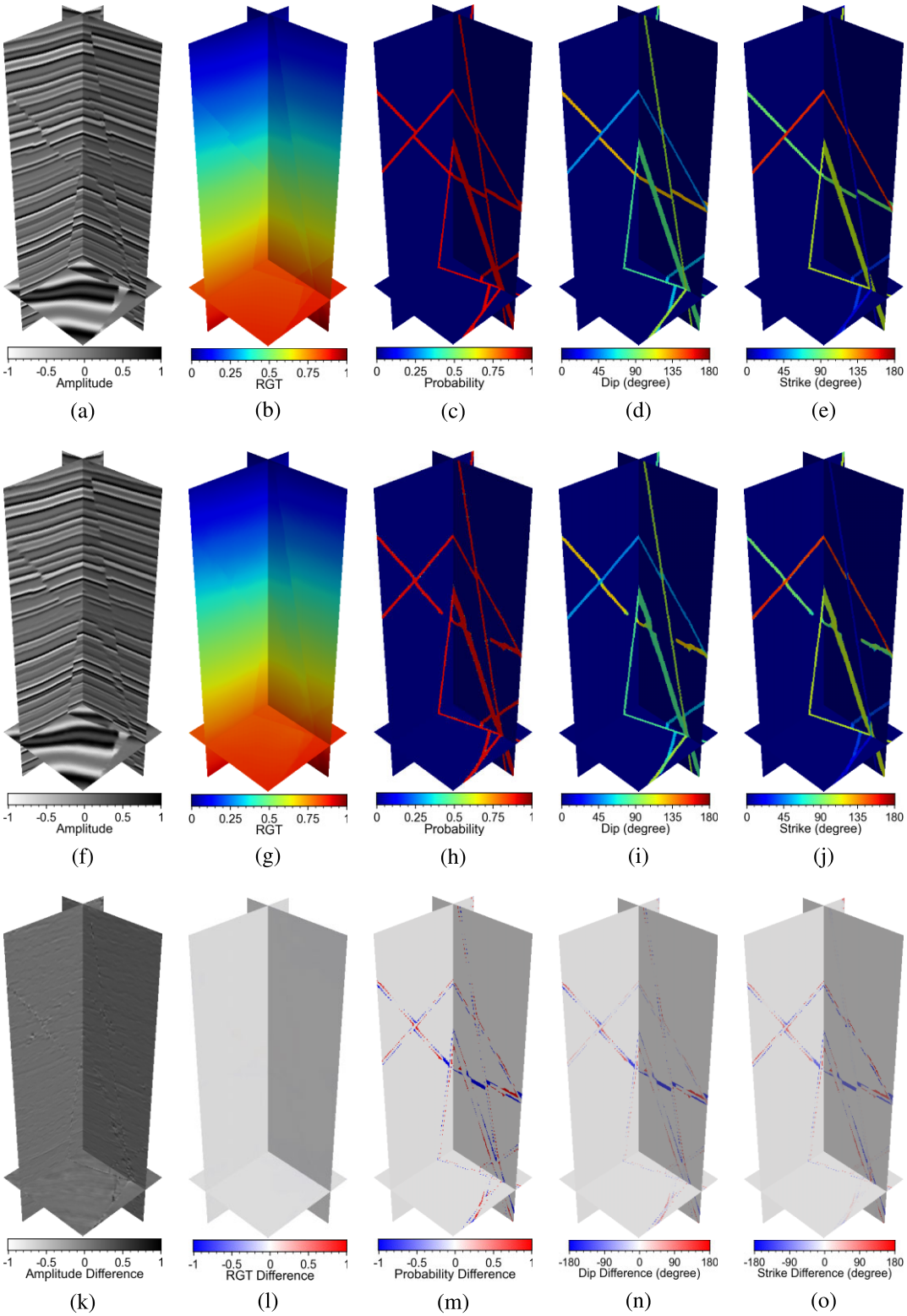

Figure 10. (a–e) DHR, RGT, fault probability, dip and strike labels of the image displayed in Fig. 9. (f–j) Inferred attributes using MTI-Net and (k–o) the differences between the inferred and ground-truth attributes.

2.3 Training data preparation and training

Generating highly realistic, high-quality seismic images and their corresponding labels is critical to training a supervised NN, particularly for geophysical applications (e.g. Zhang *et al.* 2022). In this work, we intend to train MTI/MTR-Net using synthetic image-label pairs only and apply the trained model to field data without additional training, therefore the quality of training data is particularly important. The procedure for generating the training data resembles those for fault detection (Wu *et al.* 2019a; Gao

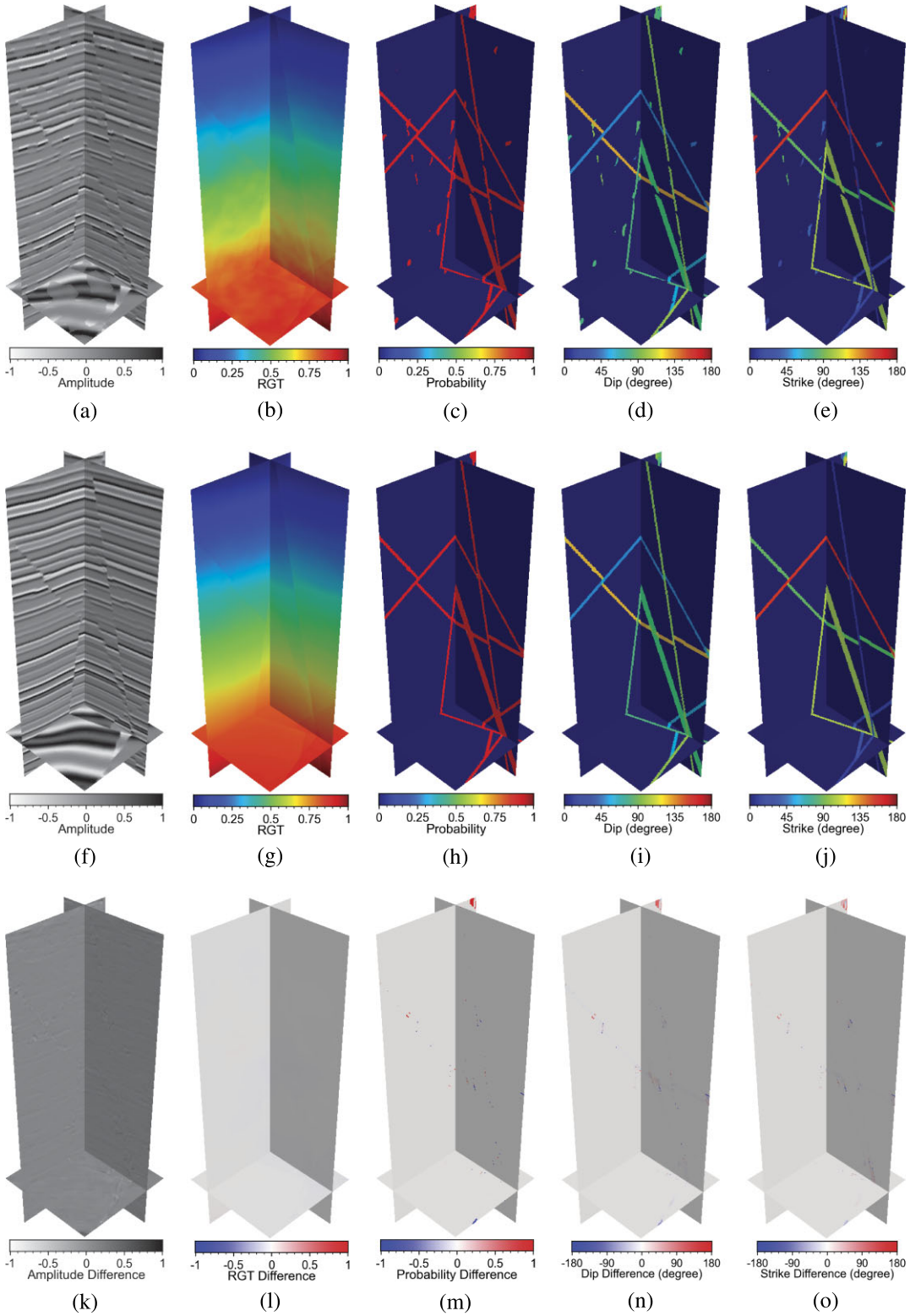

Figure 11. (a–e) Inputs to and (f–j) outputs from MTR-Net corresponding to the example image in Fig. 9. The labels are displayed in Figs 10(a)–(d). Panels (k–o) display the differences between the MTR-Net results and labels.

et al. 2022a, c) and RGT estimation (Bi *et al.* 2021). We provide the detailed procedure for generating the data-label pairs in Appendix B.

We generate a total of 5000 input-label pairs as the training set with an additional 100 input-label pairs as the validation set for training 2-D MTI/MTR-Net. The dimension of the input image and labels is $N_z \times N_x = 360 \times 256$. For 3-D training, we perform two trainings to test the performance of MTI/MTR-Net, each with a total of 2000 input-label pairs as the training set with an additional 100 input-label pairs as the validation set. The dimensions of the

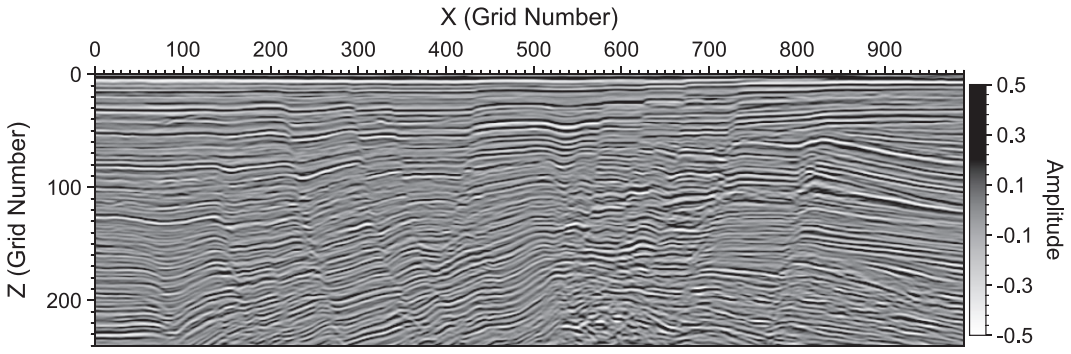

Figure 12. A 2-D seismic image sliced from the 3-D Opunake image.

input image and labels in these two trainings are $N_z \times N_y \times N_x = 256 \times 128 \times 128$ gridpoints and $N_z \times N_y \times N_x = 360 \times 128 \times 128$ gridpoints, respectively. For both 2-D and 3-D cases, the training and validation sets do not overlap, thus ensuring an unbiased validation.

We implement both 2-D and 3-D MTI/MTR-Nets using PyTorch (Paszke *et al.* 2019) under the interface wrapper of PyTorch Lightening (Falcon 2019). We train both 2-D and 3-D MTI/MTR-Nets using an Adam optimizer (Kingma & Ba 2017) with an initial learning rate of 0.5×10^{-4}, and reduce the learning rate by 10 times when the training loss does not change after ten epochs. We train both 2-D and 3-D MTI/MTR-Nets using a batch size of eight on a total of eight NVIDIA A100 graphics processing unit (GPU) cards.

During the trainings, we record the loss convergence associated with every attribute. Fig. 3(a) displays the loss convergence curves associated with the training (solid curves) and validation (dashed curves) of 3-D MTI-Net. We observe that all losses converge, with a similar level of final loss for different learning tasks under the weighting strategy defined in eq. (1). We also find a mild level of overfitting associated with RGT and fault attributes learning. Our validation shows that such mild overfitting does not obviously degrade the generalization of MTI-Net. For the losses associated with MTR-Net displayed in Fig. 3(b), we observe a reduced overfitting effect where the training losses are close to the respective validation losses. This improvement is probably due to the fact that MTR-Net maps multiple inputs to multiple outputs, and the inputs and outputs are of the same nature (i.e. from DHR, RGT, fault attributes to DHR, RGT and fault attributes), and the entire NN may be close to an identity mapping NN to some extent. By contrast, MTI-Net maps an image to DHR, RGT and fault attributes. Such a multitask learning is challenging and may impact MTI-Net's generalization capability.

It is worth noting that in this work, we use only synthetic images and labels to train MTI/MTR-Net, but this strategy may not be sufficient in some cases. For example, we only consider faults with dip angles in the range of [50°, 130°]. Consequently, special types of faults such as low-dip thrust faults cannot be well identified and characterized. We do not include magma intrusion or salt bodies in the training data either, which can be difficult to model with our current geological modelling algorithm. Angular unconformity that may terminate a fault path or surface is not included in the training data. In such cases, jointly using field-data labels in addition to synthetic data labels may improve the fidelity of training and generalization. However, we remark that it may be challenging to prepare field-data labels, particularly for MTI/MTR-Net, because one needs to prepare five labels instead of one as in single-task fault detection, RGT estimation or image denoising. How to improve the quality of synthetic training data deserves a dedicated investigation.

3 RESULTS

We validate the efficacy of our MTI/MTR-Net using several synthetic and field data images.

3.1 Synthetic images

We first validate the trained MTI-Net using a synthetic image in the validation set. Fig. 4 shows a noisy synthetic image sampled by 360 × 256 gridpoints. Figs 5(a)–(d) display the ground-truth DHR image, RGT, fault probability and fault dip, respectively. Figs 5(e)–(h) display the DHR, RGT and fault attributes generated by MTI-Net, while Figs 5(i)–(l) display the differences between the MTI-Net results with the ground truth. The DHR image estimated by MTI-Net is close to the ground truth image. All the lateral discontinuities are well revealed, and random noise is suppressed. The RGT difference indicates that the inferred RGT is close to the ground truth as well. For the fault attributes, we observe that MTI-Net reveals all target faults with correct dip angle, except for several interlacing faults to the lower right of the image centre and around $(X, Z) = (180, 100)$. We also observe in the fault attributes difference images that the inferred faults are narrower than the ground truth. This inconsistency, however, does not affect the accuracy of the spatial location and dip angle of the revealed faults.

Fig. 6 displays an example of validating MTR-Net. The labels for MTR-Net are same with those of MTI-Net as displayed in Figs 5(a)–(d). Figs 6(a)–(d) display the data (i.e. the inputs). These data, generated by the procedure detailed in Appendix B, are created purposely with 'noise'. For instance, randomly broken faults and extra small faults in Figs 6(c) and (d) are created to mimic fault inference results frequently observed in the single-pass MTI-Net (and many other ML-based fault detection methods) inference results. Figs 6(e)–(h) show the output generated by the trained MTR-Net. We also display the difference between the output and the ground truth in Figs 6(i)–(l). The accuracy of the DHR and RGT are similar with that associated with MTI-Net, while the recovery of the two fault attributes are better than that associated with MTI-Net. The improvement is anticipated, because the inputs to MTR-Net are already close to ground truth, except that some faults are purposely broken and some noise is added.

The above example on a synthetic image shows that the trained MTI-Net and MTR-Net can perform multitask inference and refinement as expected. We now use another example to investigate whether MTI/MTR-Net can achieve improved accuracy and structural consistency among the inferred attributes (DHR, RGT and faults attributes) than U-MTI (i.e., MTI based on the conventional

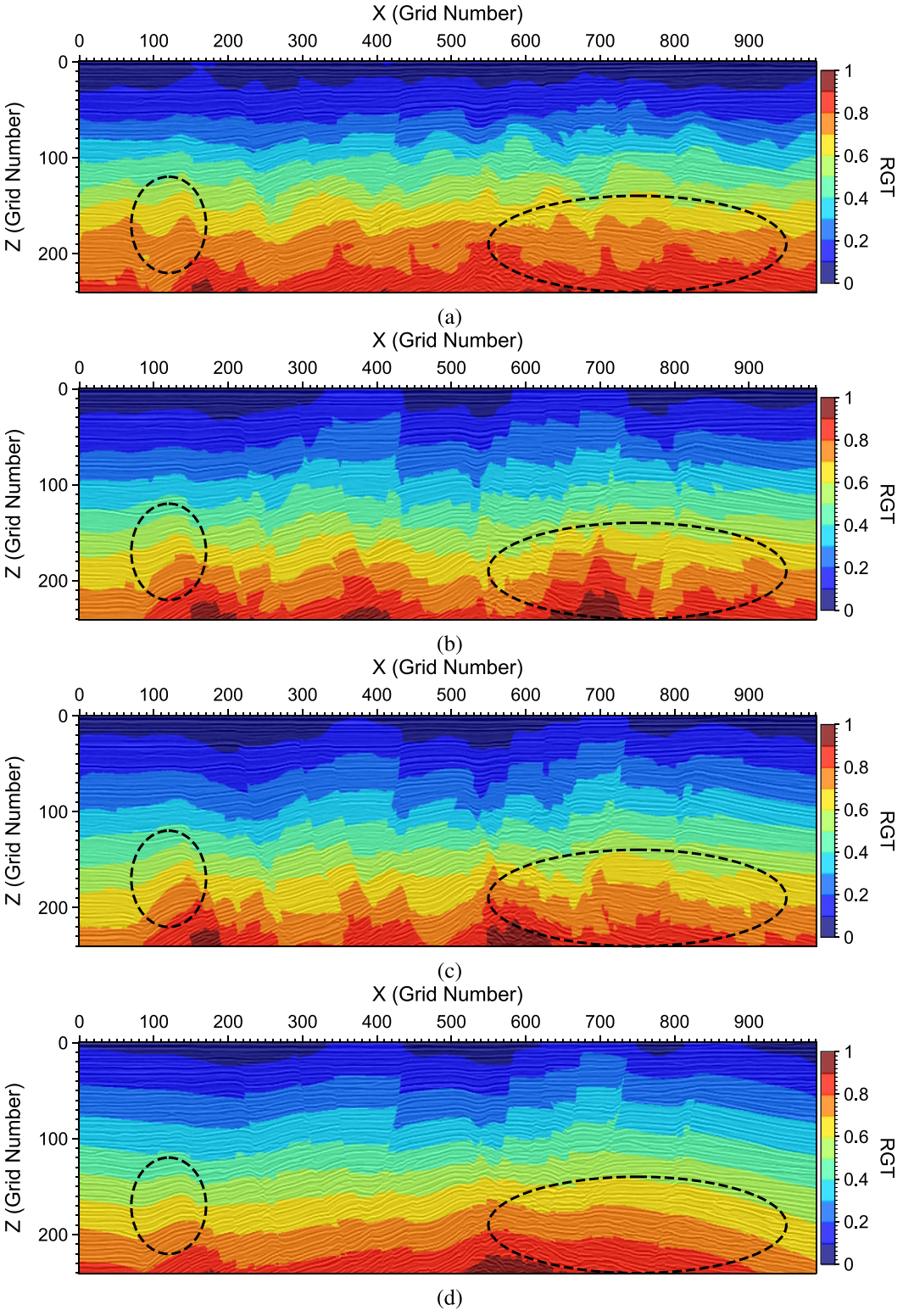

Figure 13. RGT estimated using (a) U-MTI, (b) STI, (c) MTI-Net and (d) MTR-Net after four iterations with the MTI-Net output as the input at the first iteration. The background images are DHR images estimated using the above respective methods.

U-Net structure, see Appendix A for details) and STI (i.e., single-task inference, where we turn off two of the decoder branches of our MTI during training and inference). Fig. 7 displays a noisy synthetic image sampled by 256 × 256 gridpoints. Figs 8(a) and (b) display the RGT and fault dip labels overlying on the corresponding DHR label. For better visualization, we display the RGT in filled contours ranging from 0 to 1 with a regular interval of 0.1. We display only part of the fault dip in the horizontal di-

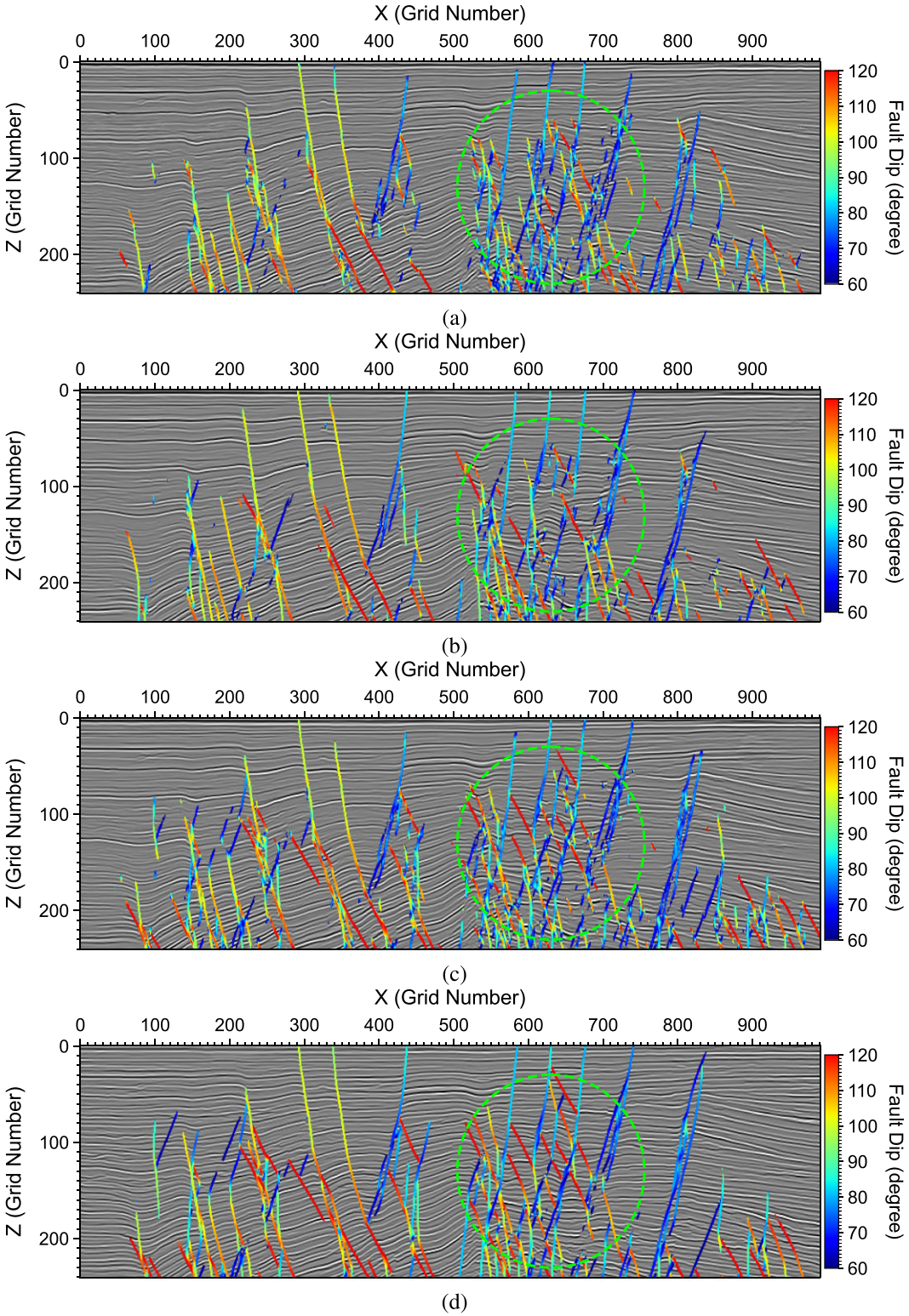

Figure 14. Fault dip estimated using (a) U-MTI, (b) STI, (c) MTI-Net and (d) MTR-Net after four iterations with the MTI-Net output as the input at the first iteration. The background images are DHR images estimated using the above respective methods.

rection instead of the whole image to highlight the fault on the right.

Figs 8(c) and (d) show the RGT and fault dip inference results overlying on the inferred DHR image, all generated by U-MTI. It is evident that the RGT inference is far away from the ground truth (Fig. 8a), nor it is consistent with the estimated DHR image. For the fault dip estimation, we find that although the U-MTI correctly inferred the path of the faults from the noisy image, there are notable variations along the fault on the right. Since we set straight faults in the synthetic image, the dip should be constant within a fault. These

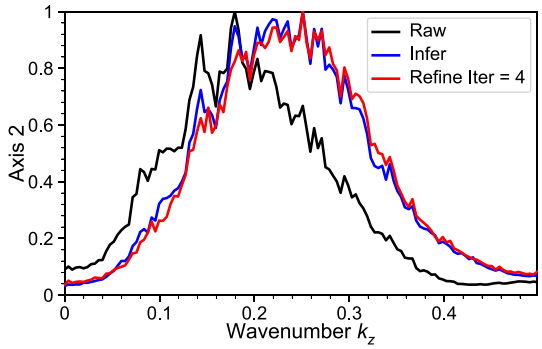

Figure 15. A comparison among the vertical wavenumber k_z spectra of the raw (black curve) and DHR (blue curve) images obtained using MTI-Net and MTR-Net at the fourth iteration (red curve).

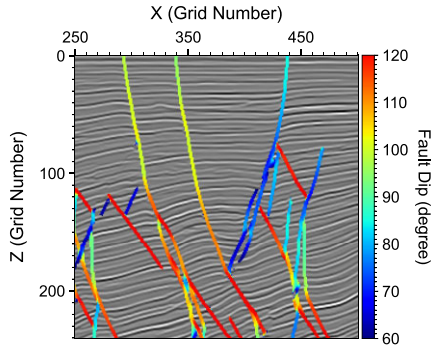

Figure 16. A zoom-in view of the fault dip displayed in Fig. 14(d).

inaccuracies are probably due to the limited size of the receptive field in the conventional U-Net architecture.

Figs 8(e) and (f) display the inference results generated by STI. The nested residual U-Net structure has an improved receptive field and therefore both the RGT and fault dip inference results are closer to the ground truth than those generated by U-MTI, even if these results are generated in a single-task manner. However, we still observe inconsistency between the RGT and DHR, particularly on the right part of the image annotated by three black arrows. The fault dip estimation is more consistent along the fault path compared with that estimated by U-MTI. However, we find some level of variations along the path of the fault on the right.

By contrast, Figs 8(g) and (h) display the RGT and fault dip inference results obtained with our MTI-Net. An evident improvement is that the RGT estimation is closer to the ground truth and importantly, is more consistent with DHR than those provided by U-MTI and STI. We refine the results using MTR-Net and display the refinement results at the third iteration in Figs 8(i) and (j), which shows the best RGT-DHR consistency among all the four five results, and the variations along the fault path are minimal.

We further validate the efficacy and accuracy of our MTI/MTR-Net using a 3-D synthetic image in the validation set displayed in Fig. 9, which has mild noise and several faults. Figs 10(a)–(e) display the labels (i.e. the ground truth), including the DHR image, RGT, fault probability, dip angle and strike angel of this image. Figs 10(f)–(j) display the inferred attributes using MTI-Net, while Figs 11(k)–(o) display the differences between the inferred and ground truth attributes. As in the 2-D synthetic image case, all the attributes are recovered with good accuracy except that the fault widths are not fully recovered. Such inaccuracy, however, does not essentially affect characterization of the most important properties (i.e. spatial location, dip and strike) of faults.

Similarly with the 2-D case, we verify the efficacy and accuracy of the trained MTR-Net in Fig. 11 based on an image in the validation set, where we display the inputs in Figs 11(a)–(e), the outputs generated by MTR-Net in Figs 11(f)–(j), and their differences in Figs 11(k)–(o). We observe a good recovery of all the attributes.

3.2 Opunake field data image

We then validate MTI/MTR-Net using a 2-D seismic image sliced and resampled from the 3-D Opunake image volume (SEG 2020). The selected image is sampled by $N_z \times N_x = 251 \times 1001$ regularly spaced gridpoints. We observe several major faults and numerous small-scale faults in the image. The complexity of these faults makes fault identification, RGT estimation and image enhancement challenging, particularly for the centre-bottom part of the image.

We first perform multitask inference on this image using U-MTI, STI and our MTI-Net. Figs 13(a)–(d) display the RGT image in the form of contours inferred using U-MTI, STI, MTI-Net and refined by MTR-Net after four iterations, respectively, overlying on the inferred/refined DHR images with the above respective methods. The results resemble those in the synthetic data example displayed in Fig. 8. We find that the RGT image produced by U-MTI displayed in Fig. 13(a) has evident inconsistency with the DHR image in both shallow and deep regions of the model. The RGT images produced by STI displayed in Fig. 13(b) and by MTI-Net in Fig. 13(c) are more consistent with their respective DHR images compared with that produced using U-MTI. However, there are some lateral variations in the deep region of the model that do not correlate well with their respective DHR images, or that are difficult to interpret, as annotated by the black dashed ellipses in Figs 13(b) and (c). MTR-Net refines the MTI-Net RGT result and produces a RGT image displayed in Fig. 13(d), which has the best structural consistency with the DHR image among the four results, indicating the efficacy of MTR-Net applied to this field data image.

Figs 14(a)–(d) display the fault dip estimated using U-MTI, STI, MTI-Net and MTR-Net after four iterations with the MTI-Net output as input at the first iteration. The fault dip estimation results are plotted over the DHR images obtained using the above respective methods. Overall, all methods reveal numerous major and small-scale faults from the input image, and estimate plausible dip angles along the fault paths. The U-MTI result in Fig. 14(a) shows numerous small-scale faults. The STI result in Fig. 14(b) improves the U-MTI result by revealing longer and more continuous faults. We use green dashed ellipses to highlight a region with high fault density, where different methods generate notably different results. In this region, there exists some inconsistency between the detected faults (and dip angles) and the DHR image, particularly around $(X, Z) = (590, 110)$ and $(X, Z) = (650, 120)$. Such inconsistency is not surprising because the DHR image and the fault dip in Fig. 14 are estimated separately using two STIs, which impose no constraint on each other. By contrast, our MTI-Net result shown in Fig. 14(c) show better consistency between DHR and RGT. However, we also observe that there are numerous small-scale faults that are difficult to interpret. We perform multitask refinement using MTR-Net and display the fault dip obtained at the fourth iteration in Fig. 14(d). Not only the continuity of faults is improved after the refinements,

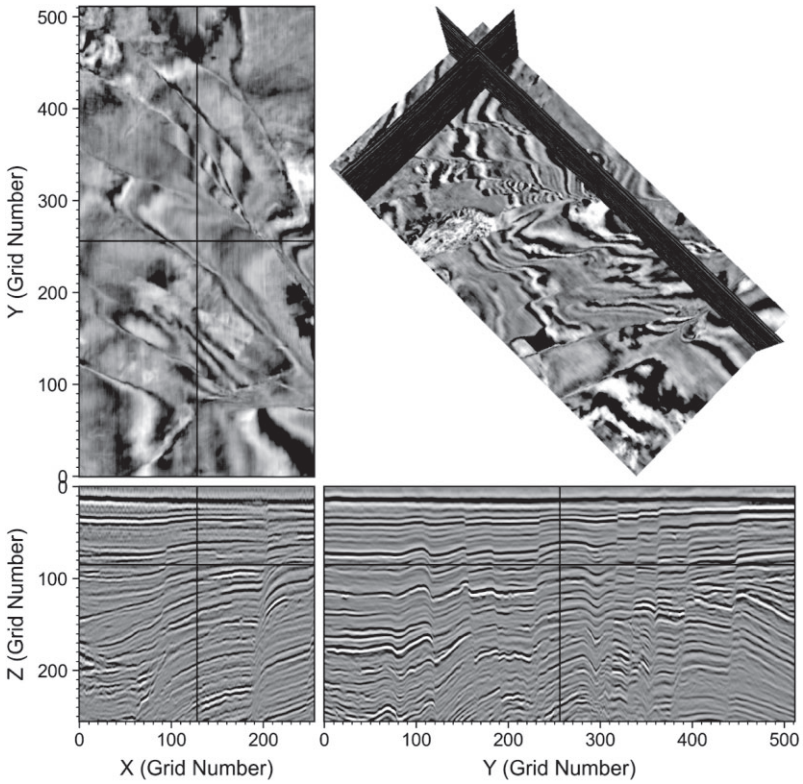

Figure 17. 3-D Opunake image volume displayed in the form of orthogonal slices.

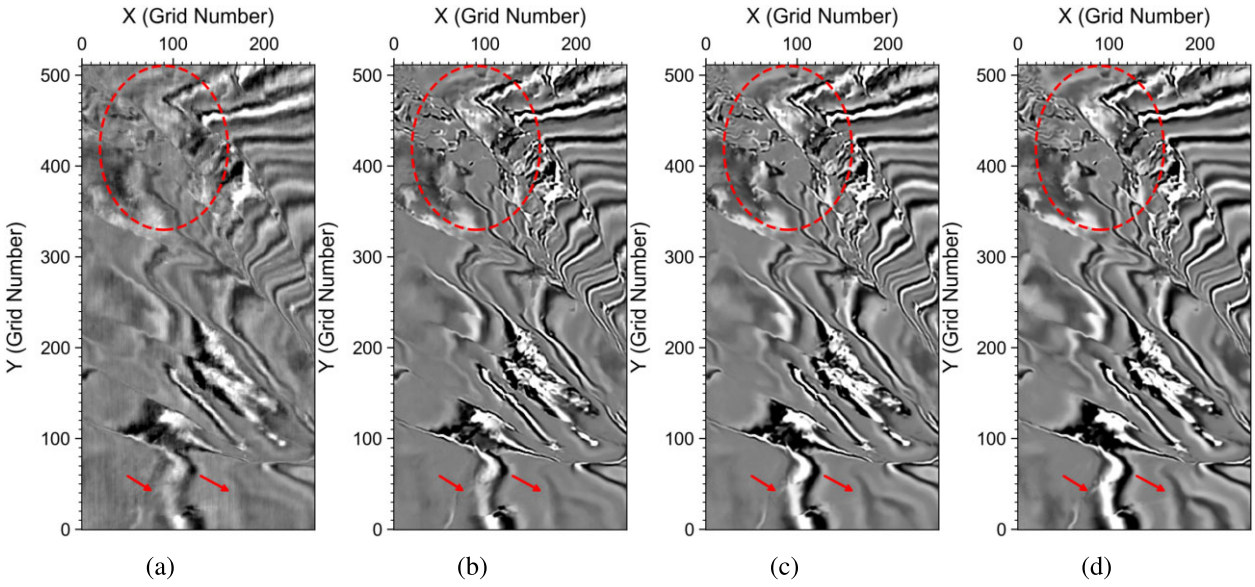

Figure 18. (a) A depth slice from the Opunake image displayed in Fig. 17. (b) Slice of the DHR image generated by MTI-Net. Slice of the DHR images refined by MTR-Net after (c) one and (d) three iterations, respectively.

random scattered faults are mostly eliminated. Moreover, the structural consistency between the DHR image and RGT is improved. In fact, we cannot find any fault that does not correspond to lateral discontinuity of reflectors on the DHR image. With such refinement results, it seems unnecessary to perform post-processing of the results to identify faults or characterize their geometrical properties with external methods (e.g. Zhao & Mukhopadhyay 2018; Wu *et al.* 2019c).

Based on our definition of a DHR image, it should have a higher vertical resolution compared with the input image to MTI-Net, but have a very close, if not the same, vertical resolution with the MTR-Net refinement DHR image. Fig. 15(c) displays the normalized wavenumber k_z spectra associated with the raw (black) image displayed in Fig. 12, the DHR image obtained by MTI-Net (blue) and the DHR image obtained by MTR-Net after four iterations (red). We observe that, as expected, the MTI-Net DHR image has

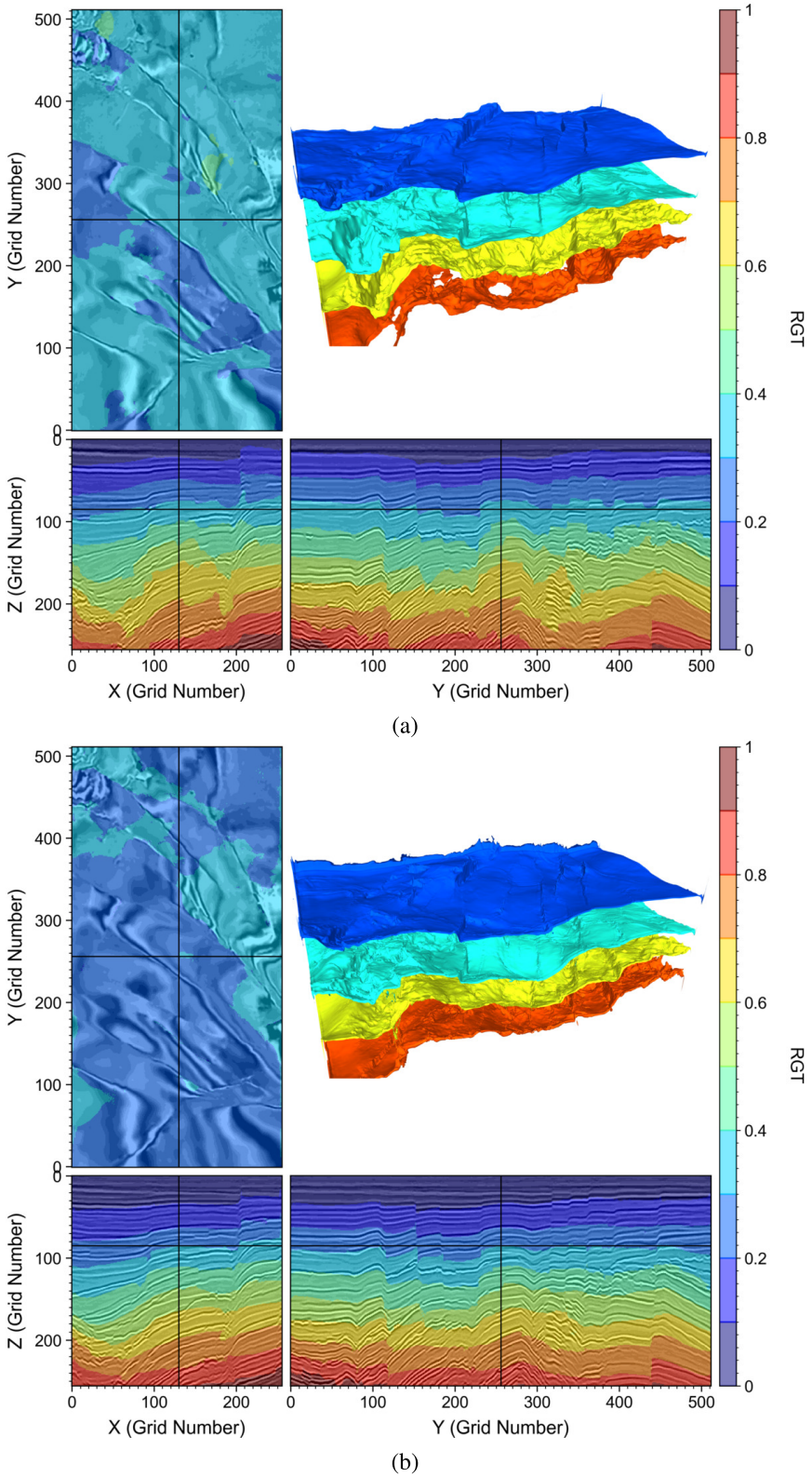

Figure 19. (a) The RGT volume inferred from the Opunake 3-D raw seismic image in Fig. 17 by MTI-Net overlying on the inferred DHR image. (b) The RGT volume refined by MTR-Net after three iterations overlying on the refined DHR image.

a higher vertical resolution than the raw seismic image, indicating the efficacy of our MTI-Net in improving the vertical resolution of a seismic image. Meanwhile, the spectrum of the MTR-Net DHR image is almost identical to that of the MTI-Net DHR image, indicating that MTR-Net does not alter the wavenumber spectrum of the input image.

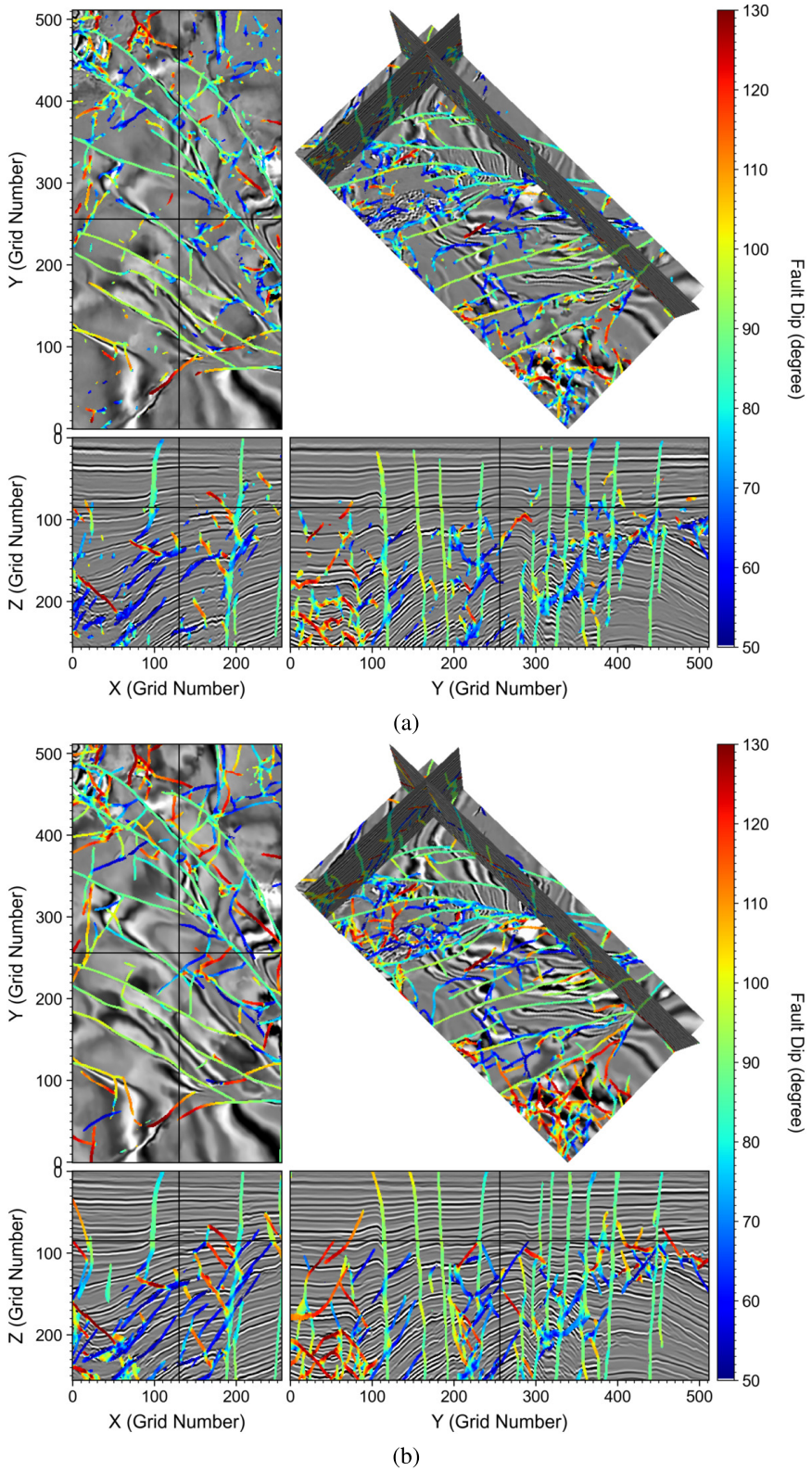

Figure 20. (a) The fault dip attribute inferred from the Opunake 3-D raw seismic image in Fig. 17 by MTI-Net overlying on the inferred DHR image. (b) The fault dip attribute refined by MTR-Net after three iterations overlying on the refined DHR image.

Note that our MTI/MTR-Net can capture subtle fault attribute variations along a fault path or surface, even though in the training data, a certain fault attribute is set to be constant along a fault path or surface surface. Fig. 16 displays a zoom-in view of the detected fault dip in the centre part of the Opunake 2-D image, which shows variations of fault dip angles along individual fault paths. This capability of MTI/MTR-Net is crucial to characterize realistic geological fault systems, because realistic fault paths and

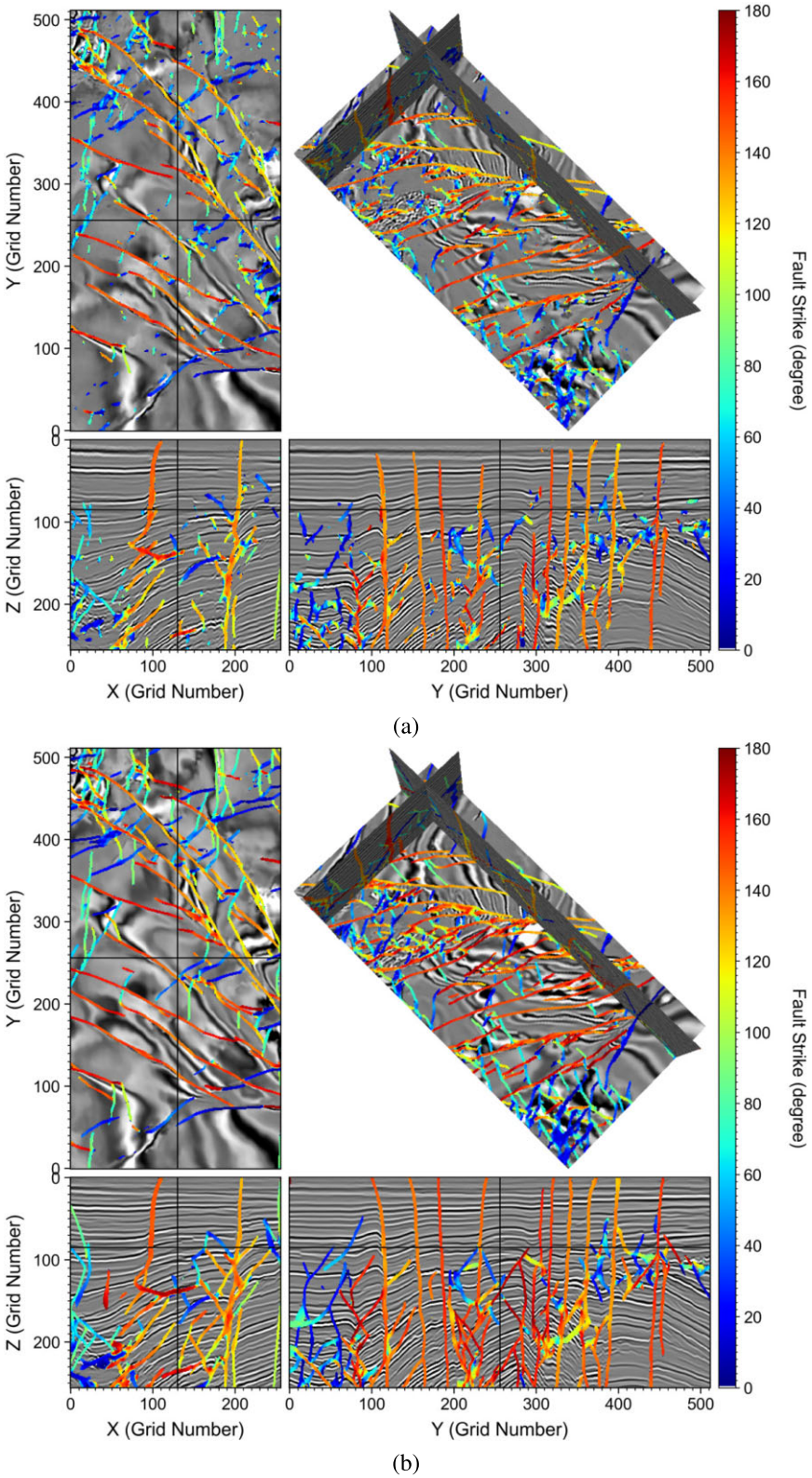

Figure 21. (a) The fault strike attribute inferred from the Opunake 3-D raw seismic image in Fig. 17 by MTI-Net overlying on the inferred DHR image. (b) The fault strike attribute refined by MTR-Net after three iterations overlying on the refined DHR image.

surfaces can be smoothly curved or even irregularly shaped, not necessarily straight or planar. The success demonstrates the validity of using synthetic seismic images and geological systems to train MTI/MTR-Net and apply them to field seismic images.

We then use the full 3-D image volume to validate our MTI/MTR-Net. Due to the memory cost associated with running MTI/MTR-Net, we resample the 3-D image volume with $N_z \times N_y \times N_x = 256 \times 512 \times 256$ regularly spaced gridpoints.

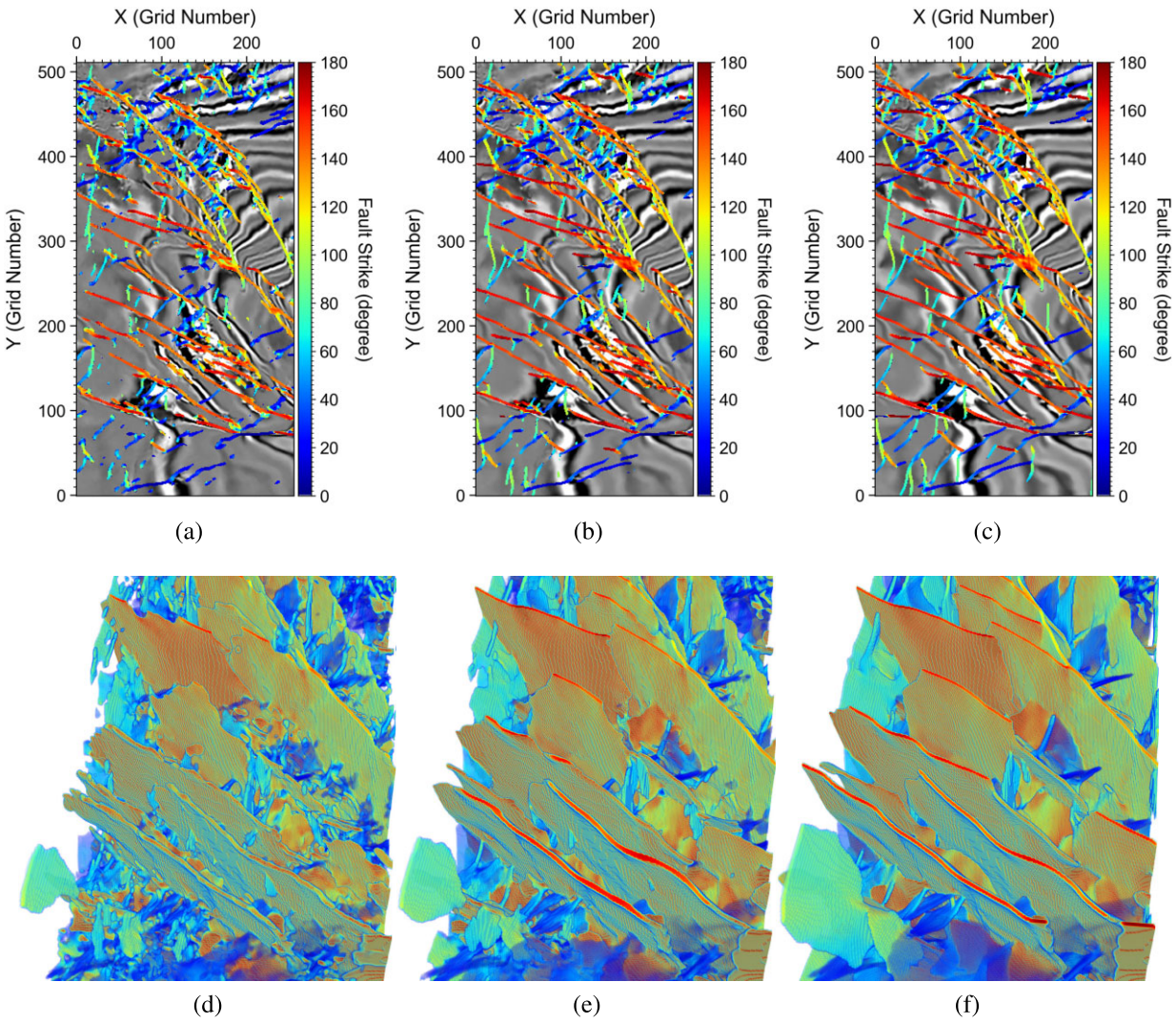

Figure 22. Depth slices of the fault strike estimated using (a) MTI-Net and MTR-Net at the (b) first iteration and (c) third iteration, respectively. Volume rendering of the estimated fault strike produced using (d) MTI-Net and MTR-Net at the (e) first iteration and (f) third iteration, respectively. The rendering is viewed from the top of the volume.

Fig. 17 displays three orthogonal slices from the 3-D image volume. The vertical slices indicate that there are multiple steeply dipping faults, while the horizontal slice shows that these faults are mostly curved and span the entire image. Although the image volume seems of high imaging quality, we still observe possible survey system footprint on the horizontal slice (X–Y slice) appearing as parallel stripes in addition to some random noise distributing in the entire volume.

We use the trained 3D MTI-Net to simultaneously infer RGT, DHR and three fault attributes from the image, and refine the outputs using MTR-Net. Fig. 18(a) displays a depth slice of the raw Opunake image, while Figs 18(b)–(d) display the DHR images generated by MTI-Net and further refined by MTR-Net after one and three iterations, respectively. Overall, the quality of these images is notably improved by MTI-Net and MTR-Net. The coherent stripe noise on the raw image is largely eliminated by MTI-Net. Structures highlighted by the red ellipses are enhanced where faults become clearer. The subtle structures pointed by the red arrows in Fig. 18(a) become well resolved in the MTI-Net inferred DHR image and even further refined by MTR-Net in Figs 18(c) and (d). MTI-Net and MTR-Net thus improve the geological interpretability of the low-resolution, noisy seismic image.

Fig. 19(a) displays the RGT overlying on the DHR image, both inferred by MTI-Net. We observe that the RGT accurately captures both the variations of sedimentary horizons and lateral discontinuities of reflectors, and the displacement of the RGT volume in the vertical direction well matches those of the reflectors. Fig. 19(b) displays the RGT overlying on the DHR image refined by the MTR-Net after three iterations. The structural consistency between the refined RGT and DHR is improved by three iterations of MTR-Net refinement.

Figs 20(a) and 21(a) display the fault dip and strike attributes overlying on the DHR image, both inferred by MTI-Net. The faults are well correlated with the lateral discontinuities of the reflectors. In particular, we observe from the horizontal slice that the inferred fault strike accurately captures fault strike variations within each major fault. The result further validates that, even though MTI-Net is trained by synthetic data where the fault strike is constant within each fault, in the inference phase it can accurately estimate variations of the strike angle in faults. Figs 20(b) and 21(b) show the fault dip and strike attributes after three iterations of refinement using MTR-Net. We find that MTR-Net eliminates most of the scattered small faults in the MTI-Net result, resulting in improved geological interpretability of the fault attributes. In particular, we display the

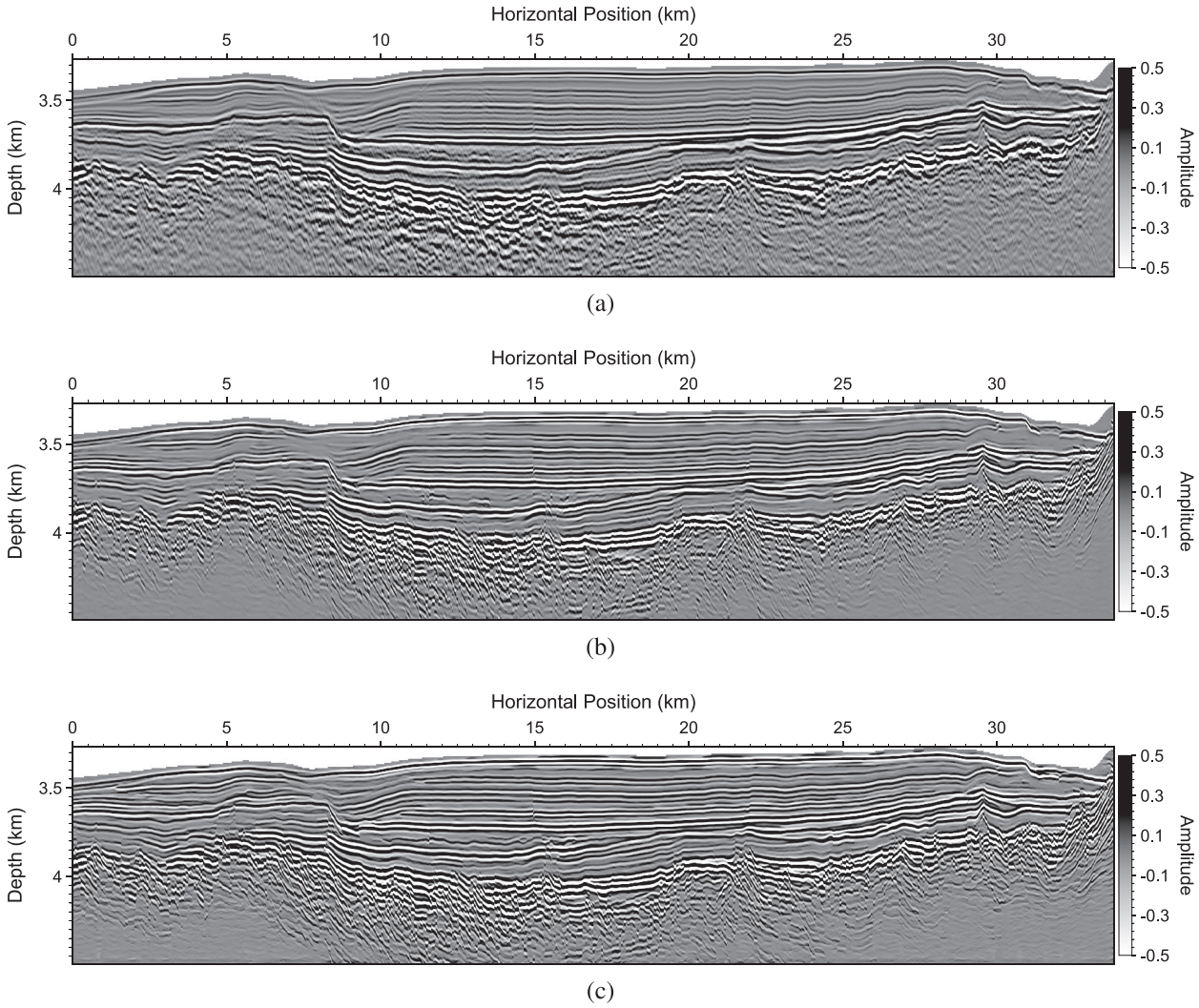

Figure 23. (a) The RTM image generated based on Shatsky Rise field seismic data for validating MTI/MTR-Net. The DHR image (b) inferred by MTI-Net and (c) refined by MTR-Net after three iterations.

depth slices extracted from the MTI-Net inferred fault strike map and MTR-Net refined fault strike maps after one and three iterations in Figs 22(a)–(c), respectively. These three slices correspond to a different depth from those in Figs 20 and 21. Moreover, we display the volume rendering of the fault strike maps generated by MTI-Net and MTR-Net after one and three iterations of refinement in Figs 22(d)–(f), respectively. We observe notably improved fault surface continuity after the MTR-Net refinement compared with that of the MTI-Net results. The comparison demonstrates the efficacy and accuracy of our MTI/MTR-Net applied to this field data image.

3.3 Shatsky Rise field data image

In the last example, we use a seismic image generated by marine multichannel seismic data acquired at Shatsky Rise, northwest Pacific Ocean (Sager *et al.* 2013, 2016), to demonstrate the efficacy of MTI/MTR-Net in delineating complex subsurface features to assist automatic interpretation.

Fig. 23(a) displays the Shatsky Rise subsurface reflectivity image produced using a wavefield-separation-based reverse-time migration (RTM) method (Fei *et al.* 2015; Chi *et al.* 2021). The displayed image is cropped from a much larger RTM image for demonstration. The selected region is approximately 34 km in the horizontal direction and approximately 1.25 km in the depth direction. The image is sampled by 328 gridpoints in the depth direction and 3004 gridpoints in the horizontal direction. We observe well-developed sedimentary layers on top of a highly faulted and fractured base at the depth of approximately 4 km. Below the base, we observe only a few continuous reflectors, indicating a strikingly different geology. Previous marine geophysical studies infer that the interface might be the boundary between the sediments and the massive igneous base below (Sager *et al.* 2016).

Figs 23(b) and (c) display the DHR images inferred by MTI-net and refined by MTR-Net after three iterations, respectively. Because the original migration image in Fig. 23(a) is already of high imaging quality, we observe limited image quality improvement by the DHR images in Figs 23(b) and (c) except that the vertical resolution is improved. Nevertheless, we still observe that MTI-Net and MTR-Net reduce random noise around the igneous rock base.

Figs 24(a) and (b) display the inferred and refined fault dip maps by MTI-Net and by MTR-Net after three iterations, respectively. The faults detected by MTI-Net are in good consistency with

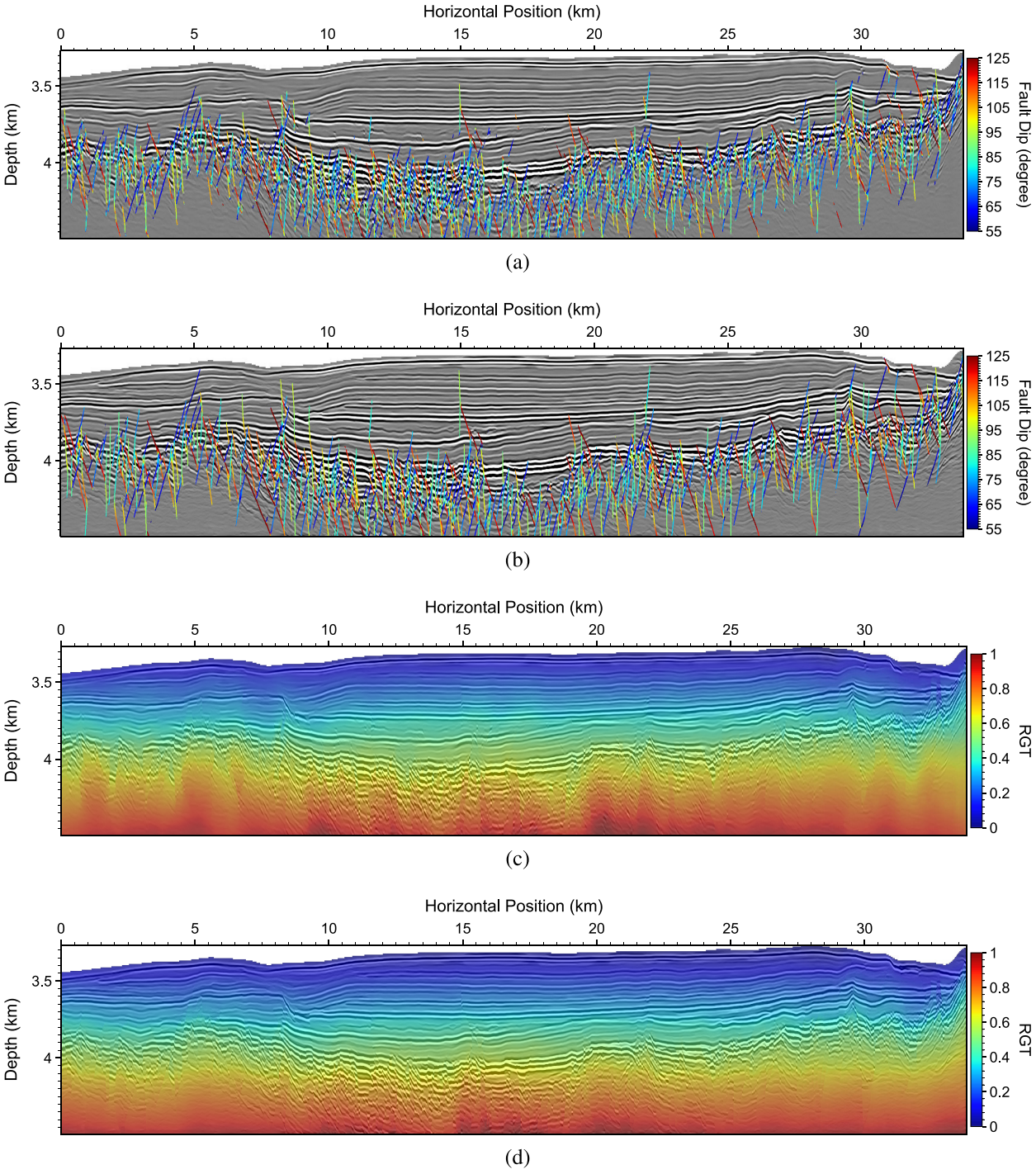

Figure 24. The fault dip (a) inferred by MTI-Net and (b) refined by MTR-Net after three iterations. The RGT (c) inferred by MTI-Net and (d) refined by MTR-Net after three iterations.

lateral discontinuities of reflectors in the sedimentary layers and around the igneous rock base, while MTR-Net refines the MTI-Net result by eliminating some randomly scattered small faults, as well as identifying and connecting broken faults into continuous faults. Figs 24(c) and (d) are the MTI-Net-inferred and MTR-Net-refined RGT images overlying on the DHR images, respectively. The initial estimation of RGT by MTI-Net has some notable lateral variations below approximately 4 km, some of which are not in good spatial correlation with the structures of reflectors. MTR-Net alleviates this issue by refining the RGT based on the image and fault dip. The refined RGT after three iterations of MTR-Net displayed in Fig. 24(d) is smoother, where the lateral variations more closely follow the shape of the igneous rock base, indicating a more reliable RGT estimation. This field data example further demonstrates the efficacy of MTI/MTR-Net in inferring and refining multiple subsurface attributes.

We demonstrate through the previous examples that MTI-Net can infer simultaneously multiple attributes from a single seismic image, and MTR-Net can refine the inference results in a multitask manner as well. Multitask inference and refinement can improve

the structural consistency among different attributes compared with single-task NNs. However, MTI/MTR-Net demands more memory to perform multitask inference and refinement compared with single-task NNs. For 2-D and small 3-D images, memory is not a practical issue with either CPU-RAM architecture (i.e. central processing unit and random-access memory) or GPU architecture (which uses video random-access memory, or VRAM). However, for large 3-D images, current mainstream GPUs can easily fall short of VRAM. In such a case, using a decomposition-merging strategy (e.g. Bi *et al.* 2021) seems to be a reasonable strategy. Nevertheless, the limitation is likely to be gradually mitigated with the fast development of GPUs (large-capacity VRAM in particular) in the future.

4 CONCLUSIONS

We have developed a novel multitask ML method that can simultaneously estimate multiple subsurface attributes from a seismic image. The multitask ML method consists of two components: a multitask inference neural network (MTI-Net) and a multimodal, multitask refinement neural network (MTR-Net). MTI simultaneously infers a relative geological time (RGT) volume, a denoised higher-resolution (DHR) image, and multiple fault attributes (probability, dip and strike) from a single seismic image. MTR simultaneously refines the input RGT, DHR and fault attributes and outputs 'regularized' attributes. We have described the workflow of generating high-quality synthetic seismic images and labels for training MTI/MTR-Net. We have validated the efficacy and accuracy of the trained MTI/MTR-Net using synthetic and field data images. The results show that our method, trained only with synthetic seismic images and labels, can perform simultaneous inference and refinement of RGT, DHR and multiple fault attributes for 2-D or 3-D field data seismic images with high fidelity. The multitask refinement stage can significantly reduce artefacts and noise in subsurface attributes estimated by the multitask inference stage. More importantly, the multitask learning strategy results in improved structural consistency among different outputs, thus enhancing the geological interpretability compared with single-task neural networks.

ACKNOWLEDGMENTS

The work was supported by the Center for Space and Earth Sciences (CSES) of Los Alamos National Laboratory (LANL). LANL is operated by Triad National Security, LLC, for the National Nuclear Security Administration (NNSA) of the U.S. Department of Energy (DOE) under Contract No. 89233218CNA000001. The Shatsky Rise multichannel seismic data were acquired during a research cruise led by Dr William Sager and Dr Jun Korenaga in 2010. The data were later pre-processed by Dr Yongchae Cho. The research used high-performance computing resources provided by LANL's Institutional Computing program. We thank two anonymous reviewers for their helpful comments.

DATA AVAILABILITY

The original Opunake image volume is available at the Society of Exploration Geophysicists open data repository https://wiki.seg.org/wiki/Opunake-3D. The source codes associated with this work are open-source available at https://github.com/lanl/mtl.

REFERENCES

Bi, Z., Wu, X., Geng, Z. & Li, H., 2021. Deep relative geologic time: a deep learning method for simultaneously interpreting 3-D seismic horizons and faults, *J. geophys. Res.,* **126**(9), doi:10.1029/2021JB021882.

bin Waheed, U., Haghighat, E., Alkhalifah, T., Song, C. & Hao, Q., 2021. Pinneik: eikonal solution using physics-informed neural networks, *Comput. Geosci.,* **155,** doi:10.1016/j.cageo.2021.104833.

Cheng, F., Chi, B., Lindsey, N.J., Dawe, T.C. & Ajo-Franklin, J.B., 2021. Utilizing distributed acoustic sensing and ocean bottom fiber optic cables for submarine structural characterization, *Sci. Rep.,* **11**(1), 5613.

Chi, B., Gao, K. & Huang, L., 2021. Vector elastic deconvolution migration with dual wavefield decomposition, *Geophysics,* **86**(4), S271–S282.

Cohen, I., Coult, N. & Vassiliou, A.A., 2006. Detection and extraction of fault surfaces in 3D seismic data, *Geophysics,* **71**(4), P21–P27.

Cunha, A., Pochet, A., Lopes, H. & Gattass, M., 2020. Seismic fault detection in real data using transfer learning from a convolutional neural network pre-trained with synthetic seismic data, *Comput. Geosci.,* **135,** doi:10.1016/j.cageo.2019.104344.

Di, H. & Gao, D., 2017. 3D seismic flexure analysis for subsurface fault detection and fracture characterization, *Pure appl. Geophys.,* **174**(3), 747–761.

Di, H., Shafiq, M. & AlRegib, G., 2018. Patch-level MLP classification for improved fault detection, in *SEG Technical Program Expanded Abstracts 2018,* pp. 2211–2215, Society of Exploration Geophysicists.

Di, H., Li, Z. & Abubakar, A., 2022. Using relative geologic time to constrain convolutional neural network-based seismic interpretation and property estimation, *Geophysics,* **87**(2), IM25–IM35.

Falcon, W., 2019. Pytorch lightning, *GitHub,* **3,** Available at: https://github.com/PyTorchLightning/pytorch-lightning.

Fei, T.W., Luo, Y., Yang, J., Liu, H. & Qin, F., 2015. Removing false images in reverse time migration: The concept of de-primary, *Geophysics,* **80**(6), S237–S244.

Gao, K., Huang, L. & Cladouhos, T., 2021. Three-dimensional seismic characterization and imaging of the Soda Lake geothermal field, *Geothermics,* **90,** 101996. 10.1016/j.geothermics.2020.101996.

Gao, K., Donahue, C., Henderson, B.G. & Modrak, R.T., 2022a. Deep-learning-guided high-resolution subsurface reflectivity imaging with application to ground-penetrating radar data, *Geophys. J. Int.,* **233**(1), 448–471.

Gao, K., Donahue, C.M., Henderson, B.G. & Modrak, R.T., 2022b. High-fidelity GPR image super-resolution via deep-supervised machine learning, *Expanded Abstracts of Second International Meeting for Applied Geoscience & Energy,* pp. 2045–2049.

Gao, K., Huang, L. & Zheng, Y., 2022c. Fault detection on seismic structural images using a nested residual U-Net, *IEEE Trans. Geosci. Remote Sens.,* **60,** 1–15. 10.1109/tgrs.2021.3073840.

Gao, K., Huang, L., Zheng, Y., Lin, R., Hu, H. & Cladohous, T., 2022d. Automatic fault detection on seismic images using a multiscale attention convolutional neural network, *Geophysics,* **87**(1), N13–N29.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y., 2014. Generative adversarial nets, *Adv. Neural Informat. Process. Syst.,* **27,** 10.48550/arXiv.1406.2661.

He, K., Zhang, X., Ren, S. & Sun, J., 2016. Deep residual learning for image recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 27-30 June 2016, Las Vegas, NV, USA, 770–778, IEEE.

Ioffe, S. & Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift, in *ICML'15: Proceedings of the 32nd International Conference on International Conference on Machine Learning,* 6–11 July 2015, Lille, France, Vol. **37,** pp. 448–456, ACM.

Jadon, S., 2020. A survey of loss functions for semantic segmentation, in *2020 IEEE Conference on Computational Intelligence in Bioinformatics*

and Computational Biology (CIBCB), pp. 27–29 October 2020, Via del Mar, Chile, pp. 1–7, IEEE.

Johnson, P.A. *et al.*, 2021. Laboratory earthquake forecasting: a machine learning competition, *Proc. Natl. Acad. Sci.,* **118**(5), e2011362118.

Kaur, H., Pham, N. & Fomel, S., 2020. Improving the resolution of migrated images by approximating the inverse Hessian using deep learning, *Geophysics,* **85**(4), WA173–WA183.

Kendall, A., Gal, Y. & Cipolla, R., 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 18–23 June 2018, Salt Lake City, UT, USA, pp. 7482–7491, IEEE.

Kingma, D.P. & Ba, J., 2017. Adam: a method for stochastic optimization, in *Proceedings of the 5th International Conference on Learning Representations, ICLR 2017,* 24–26 April 2017, Toulon, France.

Krizhevsky, A., Sutskever, I. & Hinton, G.E., 2017. Imagenet classification with deep convolutional neural networks, *Commun. ACM,* **60**(6), 84–90.

Lai, S., Xu, L., Liu, K. & Zhao, J., 2015. Recurrent convolutional neural networks for text classification, in *AAAI'15: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence,* pp. 2267–2273, AAAI Press.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P., 1998. Gradient-based learning applied to document recognition, *Proc. IEEE,* **86**(11), 2278–2324.

Leong, Z.X. & Zhu, T., 2021. Direct velocity inversion of ground penetrating radar data using GPRNet, *J. geophys. Res.,* **126**(6), e2020JB021047.

Li, J., Wu, X. & Hu, Z., 2022. Deep learning for simultaneous seismic image super-resolution and denoising, *IEEE Trans. Geosci. Remote Sens.,* **60,** 1–11.

Li, S., Yang, C., Sun, H. & Zhang, H., 2019. Seismic fault detection using an encoder-decoder convolutional neural network with a small training set, *J. geophys. Eng.,* **16**(1), 175–189.

Lomask, J., Guitton, A., Fomel, S., Claerbout, J. & Valenciano, A.A., 2006. Flattening without picking, *Geophysics,* **71**(4), P13–P20.

Lou, Y., Zhang, B., Lin, T. & Cao, D., 2020. Seismic horizon picking by integrating reflector dip and instantaneous phase attributes, *Geophysics,* **85**(2), O37–O45.

Marfurt, K.J., Kirlin, R.L., Farmer, S.L. & Bahorich, M.S., 1998. 3-D seismic attributes using a semblance-based coherency algorithm, *Geophysics,* **63**(4), 1150–1165.

Marfurt, K.J., Sudhaker, V., Gersztenkorn, A., Crawford, K.D. & Nissen, S.E., 1999. Coherency calculations in the presence of structural dip, *Geophysics,* **64**(1), 104–111.

Mousavi, S.M., Ellsworth, W.L., Zhu, W., Chuang, L.Y. & Beroza, G.C., 2020. Earthquake transformer – an attentive deep-learning model for simultaneous earthquake detection and phase picking, *Nat. Commun.,* **11**(1), 3952. 10.1038/s41467-020-17591-w.

Murphy, K.P., 2022. *Probabilistic Machine Learning: An introduction,* MIT Press.

Paszke, A. *et al.*, 2019. Pytorch: an imperative style, high-performance deep learning library, in *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019),* Vancouver, Canada, pp. 8024–8035, eds Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. & Garnett, R., Curran Associates, Inc.

Pedersen, S.I., Randen, T., Sonneland, L. & Steen, Ø., 2005. Automatic fault extraction using artificial ants, in *SEG Technical Program Expanded Abstracts,* pp. 512–515.

Poggiagliolmi, E. & Vesnaver, A., 2014. Instantaneous phase and frequency derived without user-defined parameters, *Geophys. J. Int.,* **199**(3), 1544–1553.

Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O.R. & Jagersand, M., 2020. U^2-Net: going deeper with nested U-structure for salient object detection, *Pattern Recog.,* **106,** doi:10.1016/j.patcog.2020.107404.

Qin, Y., Chen, T., Ma, X. & Chen, X., 2022. Forecasting induced seismicity in Oklahoma using machine learning methods, *Sci. Rep.,* **12**(1), 9319.

Ronneberger, O., Fischer, P. & Brox, T., 2015. U-Net: convolutional networks for biomedical image segmentation, in *Proceedings of the Medical Image Computing and Computer-Assisted Intervention (MICCAI),* LNCS, Vol. 9351, pp. 234–241, Springer.

Ross, Z.E., Meier, M.-A. & Hauksson, E., 2018. P wave arrival picking and first-motion polarity determination with deep learning, *J. geophys. Res.,* **123**(6), 5120–5129.

Rouet-Leduc, B., Hulbert, C. & Johnson, P.A., 2019. Continuous chatter of the Cascadia subduction zone revealed by machine learning, *Nat. Geosci.,* **12**(1), 75–79.

Sager, W.W., Zhang, J., Korenaga, J., Sano, T., Koppers, A. A.P., Widdowson, M. & Mahoney, J.J., 2013. An immense shield volcano within the Shatsky Rise oceanic plateau, northwest Pacific Ocean, *Nat. Geosci.,* **6**(11), 976–981.

Sager, W.W., Sano, T. & Geldmacher, J., 2016. Formation and evolution of Shatsky Rise oceanic plateau: insights from IODP Expedition 324 and recent geophysical cruises, *Earth-Sci. Rev.,* **159,** 306–336. 10.1016/j.earscirev.2016.05.011.

SEG, 2020. *New Zealand Opunake 3D data,* SEG Wiki, https://wiki.seg.org/wiki/Opunake-3D.

Song, C., Alkhalifah, T. & Waheed, U.B., 2021. A versatile framework to solve the Helmholtz equation using physics-informed neural networks, *Geophys. J. Int.,* **228**(3), 1750–1762.

Stark, T.J., 2004. Relative geologic time (age) volume – relating every seismic sample to a geologically reasonable horizon, *Leading Edge,* **23**(9), 928–932.

Sudre, C.H., Li, W., Vercauteren, T., Ourselin, S. & Jorge Cardoso, M., 2017. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations, in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support,* pp. 240–248, ed. Cardoso, M. *et al.*, DLMIA ML-CDS 2017. Lecture Notes in Computer Science, Vol. 10553, Springer, Cham.

Sun, B. & Alkhalifah, T., 2020. Ml-descent: an optimization algorithm for full-waveform inversion using machine learning, *Geophysics,* **85**(6), R477–R492.

Ulyanov, D., Vedaldi, A. & Lempitsky, V.S., 2016. *Instance normalization: the missing ingredient for fast stylization,* ArXiv preprint (arxiv.org/abs/1607.08022).

Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D. & Van Gool, L., 2022. Multi-task learning for dense prediction tasks: a survey, *IEEE Trans. Pattern Anal. Mach. Intell.,* **44**(7), 3614–3633.

Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E., 2004. Image quality assessment: from error visibility to structural similarity, *IEEE Trans. Image Process.,* **13**(4), 600–612.

Wang, Z., Li, B., Liu, N., Wu, B. & Zhu, X., 2022. Distilling knowledge from an ensemble of convolutional neural networks for seismic fault detection, *IEEE Geosci. Remote Sens. Lett.,* **19,** 1–5.

Wu, X., 2017. Directional structure-tensor-based coherence to detect seismic faults and channels, *Geophysics,* **82**(2), A13–A17.

Wu, X. & Fomel, S., 2018. Automatic fault interpretation with optimal surface voting, *Geophysics,* **83**(5), O67–O82.

Wu, X. & Guo, Z., 2018. Detecting faults and channels while enhancing seismic structural and stratigraphic features, *Interpretation,* **7**(1), T155–T166.

Wu, X., Liang, L., Shi, Y. & Fomel, S., 2019a. FaultSeg3D: using synthetic data sets to train an end-to-end convolutional neural network for 3D seismic fault segmentation, *Geophysics,* **84**(3), IM35–IM45.

Wu, X., Liang, L., Shi, Y., Geng, Z. & Fomel, S., 2019b. Multitask learning for local seismic image processing: fault detection, structure-oriented smoothing with edge-preserving, and seismic normal estimation by using a single convolutional neural network, *Geophys. J. Int.,* **219**(3), 2097–2109.

Wu, X., Shi, Y., Fomel, S., Liang, L., Zhang, Q. & Yusifov, A.Z., 2019c. FaultNet3D: predicting fault probabilities, strikes, and dips with a single convolutional neural network, *IEEE Trans. Geosci. Remote Sens.,* **57**(11), 9138–9155.

Wu, Y. & Lin, Y., 2020. Inversionnet: an efficient and accurate data-driven full waveform inversion, *IEEE Trans. Comput. Imag.,* **6,** 419–433. 10.1109/TCI.2019.2956866

Xiong, W., Ji, X., Ma, Y., Wang, Y., AlBinHassan, N.M., Ali, M.N. & Luo, Y., 2018. Seismic fault detection with convolutional neural network, *Geophysics,* **83**(5), O97–O103.

Zhang, B., Pu, Y., Xu, Z., Liu, N., Li, S. & Li, F., 2022. Exploring factors affecting the performance of deep learning in seismic fault attribute computation, *Interpretation,* **10**(4), T619–T636.

Zhao, T. & Mukhopadhyay, P., 2018. A fault-detection workflow using deep learning and image processing, in *SEG Technical Program Expanded Abstracts,* pp. 1966–1970, Society of Exploration Geophysicists.

Zhu, J., Park, T., Isola, P. & Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks, in *2017 IEEE International Conference on Computer Vision (ICCV),* 22–29 October 2017, Venice, Italy, pp. 2242–2251, IEEE.

Zhu, W. & Beroza, G.C., 2018. PhaseNet: a deep-neural-network-based seismic arrival-time picking method, *Geophys. J. Int.,* **216**(1), 261–273.

APPENDIX A: ARCHITECTURES OF THE ResUNet BLOCKS

We display the architectures of the residual U-Net blocks that form MTI/MTR-Net in Fig. A1. Each of these residual U-Net blocks is a simplified multilevel U-Net compared with the original architecture of U-Net (Ronneberger *et al.* 2015). C in these blocks represents the number of channels output by a convolution layer (C_{out}) in the PyTorch notation (Paszke *et al.* 2019). Note that ResUNet 3 does not contain downsampling or upsampling as in ResUNet 1 and ResUNet 2, but the dilation ratios are different for different convolutional blocks in ResUNet-3.

The structure of ResUNet 1 can be written as

$$x_0 = \mathcal{C}_{k=3,N=16,d=1}(x), \tag{A1}$$

$$x_1 = \mathcal{C}_{k=3,N=16,d=1}(x_0), \tag{A2}$$

$$x_2 = \mathcal{C}_{k=3,N=32,d=1}(\mathcal{M}_2(x_1)), \tag{A3}$$

$$x_3 = \mathcal{C}_{k=3,N=64,d=1}(\mathcal{M}_2(x_2)), \tag{A4}$$

$$x_4 = \mathcal{C}_{k=3,N=256,d=2}(\mathcal{M}_2(x_3)), \tag{A5}$$

$$y_3 = \mathcal{C}_{k=3,N=64,d=1}(\mathcal{U}_2(x_4) \oplus x_3), \tag{A6}$$

$$y_2 = \mathcal{C}_{k=3,N=32,d=1}(\mathcal{U}_2(y_3) \oplus x_2), \tag{A7}$$

$$y_1 = \mathcal{C}_{k=3,N=16,d=1}(\mathcal{U}_2(y_2) \oplus x_1), \tag{A8}$$

$$y_0 = \mathcal{C}_{k=3,N=16,d=1}(y_1) + x_0, \tag{A9}$$

where x is the input feature map, $x_{0,1,2,3,4}$ and $y_{1,2,3}$ are intermediate feature maps, y_0 is the output feature maps, $\mathcal{C}_{k,N,d} = \mathcal{R} \circ \mathcal{I} \circ \text{Conv}_{k,N,d}$ represents a convolution with a kernel size of $k \times k \times k$, an output channel number of N and a dilation of d, followed by a rectified linear unit (ReLU) activation layer $\mathcal{R}$ and an instance normalization layer $\mathcal{I}$. $\mathcal{M}_2$ is a max pooling layer with a ratio of 2, $\mathcal{U}_2$ is a bilinear/trilinear upscaling with a ratio of 2 and $\oplus$ is concatenation along the channel dimension.

Similarly, ResUNet 2 can be written as

$$x_0 = \mathcal{C}_{k=3,N=32,d=1}(x), \tag{A10}$$

$$x_1 = \mathcal{C}_{k=3,N=32,d=1}(x_0), \tag{A11}$$

$$x_2 = \mathcal{C}_{k=3,N=64,d=1}(\mathcal{M}_2(x_1)), \tag{A12}$$

$$x_3 = \mathcal{C}_{k=3,N=256,d=2}(\mathcal{M}_2(x_2)), \tag{A13}$$

$$y_2 = \mathcal{C}_{k=3,N=64,d=1}(\mathcal{U}_2(x_3) \oplus x_2), \tag{A14}$$

$$y_1 = \mathcal{C}_{k=3,N=32,d=1}(\mathcal{U}_2(y_2) \oplus x_1), \tag{A15}$$

$$y_0 = \mathcal{C}_{k=3,N=32,d=1}(y_1) + x_0, \tag{A16}$$

ResUNet 3 can be written as

$$x_0 = \mathcal{C}_{k=3,N=64,d=1}(x), \tag{A17}$$

$$x_1 = \mathcal{C}_{k=3,N=64,d=1}(x_0), \tag{A18}$$

$$x_2 = \mathcal{C}_{k=3,N=128,d=2}(x_1), \tag{A19}$$

$$x_3 = \mathcal{C}_{k=3,N=128,d=4}((x_2), \tag{A20}$$

$$x_4 = \mathcal{C}_{k=3,N=256,d=8}(x_3), \tag{A21}$$

$$y_3 = \mathcal{C}_{k=3,N=128,d=4}(x_4 \oplus x_3), \tag{A22}$$

$$y_2 = \mathcal{C}_{k=3,N=128,d=2}(y_3 \oplus x_2), \tag{A23}$$

$$y_1 = \mathcal{C}_{k=3,N=64,d=1}(y_2 \oplus x_1), \tag{A24}$$

$$y_0 = \mathcal{C}_{k=3,N=64,d=1}(y_1) + x_0, \tag{A25}$$

Note that these structures are simplified U-Nets with residual connections. In a conventional U-Net (Ronneberger *et al.* 2015; Wu *et al.* 2019a), the encode/decoder at each level normally consists of two or more convolutional layers. The 'residual connection' here also slightly abuses the naming of conventional residual connection (He *et al.* 2016). In a conventional residual connection, the input feature map x is usually skip-added to y_1 to generate y_0 directly or after a simple $k = 1$ convolution.

For the purpose of completeness, the U-Net we adopt for building U-MTI is

$$x_1 = \mathcal{C}^{l=3}_{k=3,N=16,d=1}(x), \tag{A26}$$

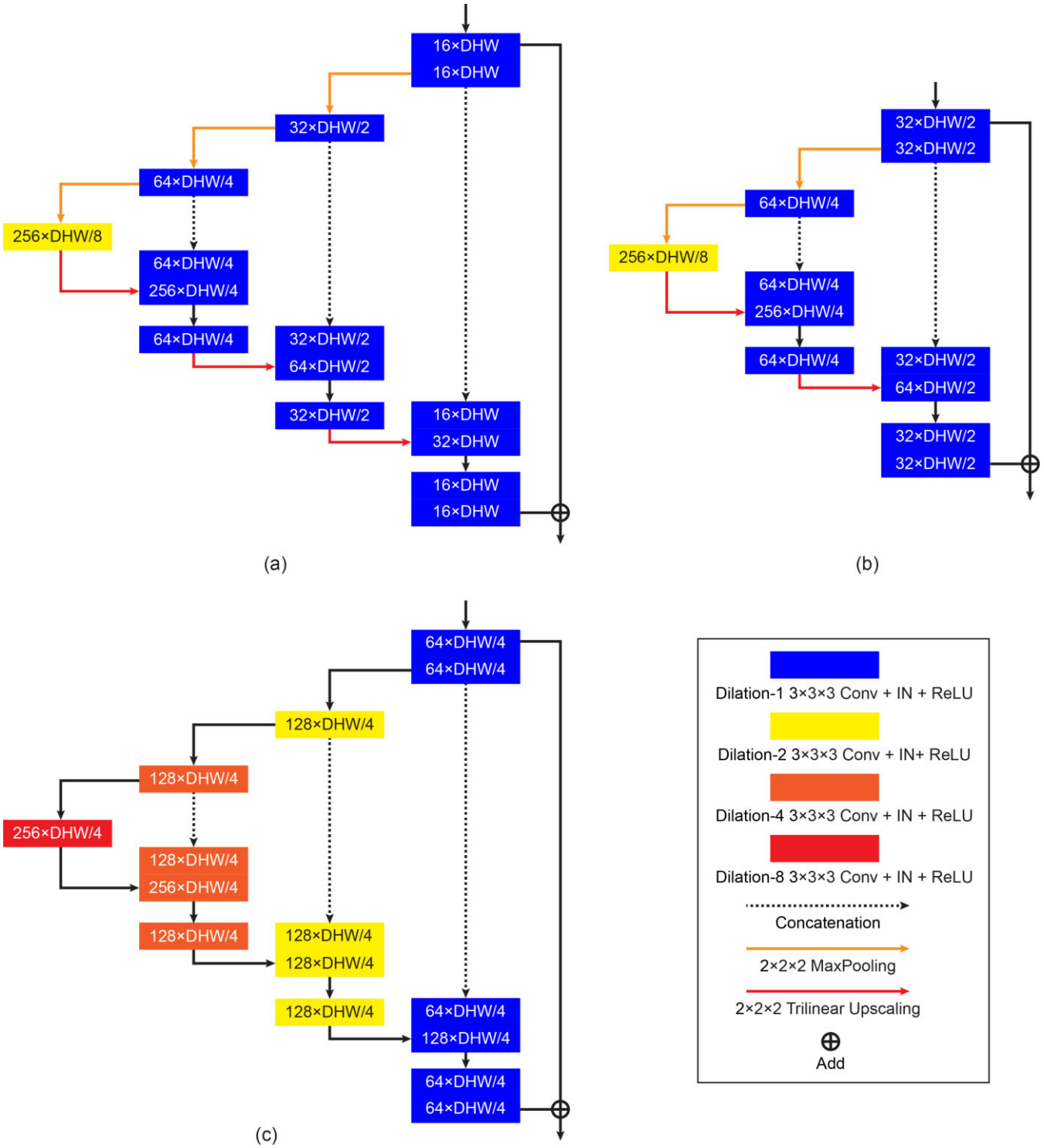

Figure A1. Architectures of (a) ResUNet 1, (b) ResUNet 2 and (c) ResUNet 3 that construct MTI/MTR-Net shown in Fig. 2. We use different colors to represent convolutional layers with different dilation ratios. '$C \times DHW/n$' means a convolutional layer with C output channels and a depth, height and width of D/n, H/n and W/n, respectively. The number of input channels can be straightforwardly determined from the the structure of the MTI/MTR-Net displayed in Fig. 2.

$$x_2 = \mathcal{C}^{l=3}_{k=3,N=32,d=1}(\mathcal{M}_2(x_1)), \tag{A27}$$

$$x_3 = \mathcal{C}^{l=3}_{k=3,N=64,d=1}(\mathcal{M}_2(x_2)), \tag{A28}$$

$$x_4 = \mathcal{C}^{l=3}_{k=3,N=512,d=1}(\mathcal{M}_2(x_3)), \tag{A29}$$

$$y_3 = \mathcal{C}^{l=3}_{k=3,N=64,d=1}(\mathcal{U}_2(x_4) \oplus x_3), \tag{A30}$$

$$y_2 = \mathcal{C}^{l=3}_{k=3,N=32,d=1}(\mathcal{U}_2(y_3) \oplus x_2), \tag{A31}$$

$$y_1 = \mathcal{C}^{l=3}_{k=3,N=16,d=1}(\mathcal{U}_2(y_2) \oplus x_1), \tag{A32}$$

where $l = 3$ represents three repeated $\mathcal{C}_{k,N,d}$. The encoder branch in MTI/MTR-Net consists of ResUNet 1, ResUNet 2 and ResUNet 3, while the decoder branch consists of ResUNet 2 and ResUNet 1. In U-MTI, however, the encoder branch consists of (x_1, x_2, x_3, x_4), while the decoder branch consists of (y_3, y_2, y_1). Additional layers/subdecoders following y_1 are same with those in MTI/MTR-Net displayed in Fig. 2.

APPENDIX B: PROCEDURE FOR GENERATING TRAINING DATA FOR MTI/MTR-Net

We design the following procedure to generate an input-label pair for MTI-Net:

(i) *Generate a random reflectivity series*: Randomly draw N_l values from a normal or uniform distribution to form a sequence r_i, where N_l is the number of reflectors, subtract its mean $\bar{r}$ and place these nonzero coefficients at random locations within $[0, N_z - 1]$ where N_z is the number of gridpoints in the vertical direction of an input image. The step is illustrated in Fig. B1.

(ii) *Generate an unfaulted image*: Generate a 3-D reflectivity image $r(x, y, z)$ using the random reflectivity model generated in Step 1, and randomly shift the traces up and down with displacement $d(x, y)$ where $d(x, y)$ is a smoothed random field. The field $d(x, y)$ can also be a combination of several 2-D Gaussian functions to mimic folds. The step is illustrated in Fig. B2.

(iii) *Generate an unfaulted RGT*: Compute the instantaneous phase $\phi(z) = \mathcal{I}[r(\cdot, \cdot, z) * w(z)]$ (Poggiagliolmi & Vesnaver 2014) where $w(z)$ is the source wavelet, $r(\cdot, \cdot, z)$ is any trace along z in $r(x, y, z)$ and '$*$' represents convolution, smooth $\phi(z)$ using a 1-D Gaussian filter with $\sigma = 5$ gridpoints, then shift the 1D RGT based on $d(x, y)$ as in Step 2 to generate a 3-D unfaulted RGT volume. The step results in a horizontally smoothly varying RGT volume, say, $R(x, y, z)$. The step is illustrated in Fig. B3.

(iv) *Random faulting*: To add N_f faults to the reflectivity image and the RGT, we assign random dips, strikes and displacements (i.e. the relative shift between the two blocks on the two sides of a fault) within predefined value ranges to the faults. For each fault f_i, we shift the block on one side of f_i for both RGT and the reflectivity image and record the fault's dip and strike angles with two independent arrays of the same size with the input image. The step is illustrated in Fig. B4.

(v) *Generate a noise-free faulted image*: We convolve the image with a point spread function generated by multiplying a wavelet $w(z)$ and a Gaussian window function with an expression of $\exp\left[-((x - x_0)^2/\sigma_x^2 + (y - y_0)^2/\sigma_y^2 + (z - z_0)^2/\sigma_z^2)/2\right]$, where σ_x, σ_y and σ_z are standard deviations of the Gaussian along x-, y- and z-axis, respectively, and (x_0, y_0, z_0) is the centre of the image volume. The step is illustrated in Fig. B5.

(vi) *Add noise*: We then add a Gaussian-smoothed random field $\eta(x, y, z)$ that was drawn from a normal or exponential distribution to the image $I(x, y, z)$ generated in Step 4 as noise. Note that we subtract the mean of the generated noise and scale it according to some predefined noise level before adding it to the image. We set $\max[\eta(x, y, z)] = \alpha \max[I(x, y, z)]$ where α is a scaling factor. The step is illustrated in Fig. B6.

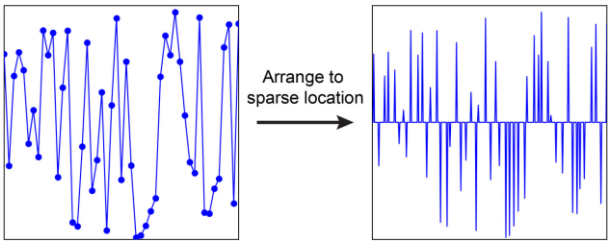

Figure B1. The procedure for generating random reflectivity.

In the above procedure, we randomly use Gaussian's first-order, second-order, or third-order derivative as the source wavelet $w(z)$ for generating different images. Because we also need a noise-free, higher-resolution image as the DHR label, therefore to generate an input-label pair, the above procedure runs twice: one for generating a low-resolution, noisy image (i.e. the input) with the peak frequency of $w(z)$ being f_0 and $\alpha \neq 0$, while the other run for generating a higher-resolution, noise-free image (i.e. the DHR image) with the peak frequency of $w(z)$ being $f_0' = 1.25 f_0$ (Gao *et al.* 2022a) and $\alpha = 0$. It is possible to use a wavelet with an even higher frequency or wider bandwidth, including the Dirac delta function (Gao *et al.* 2022b) or its band-limited version, to generate the noise-free image. Because the randomness of the faults (including the dips, strikes, rakes and displacements) directly affects the resulting image, we, therefore, use the same random seed for these two runs. Note that we set the number of faults N_f and the number of reflectors N_l to vary from image to image, thus improving the reality of the generated images. In addition, the magnitude of the added noise differs from image to image during the generation process so different images may have different levels of random noise. In this way, our training images can mimic realistic migration images with weak to strong noise and coherent artefacts. In Step 6, we randomly vary α from 0 to 0.6 for different images.

To train MTR-Net, we adopt a similar procedure to generate the image, RGT and fault attributes. However, the purpose of MTR-Net, as we described in the Methodology section, is to 'heal' the discontinuities/holes in fault surfaces, and to eliminate randomly scattered small-scale faults caused by inaccurate inference that can be difficult to interpret; in addition, we hope to refine the RGT and DHR, so that noise or artefacts in these volumes can be attenuated. To achieve this goal, we use the following strategy to generate the input (i.e. the data in the data-label pairs) for MTR-Net during the procedure of generating data-label pairs for MTI-Net.

To generate a noisy DHR, we add a low level of smoothed random noise to the DHR image of MTI-Net in addition to random scaling as in the above noisy fault attribute and RGT volume generation. In this way, we obtain a noisy DHR image that has the same wavenumber spectrum with the DHR label of MTI-Net. An example noisy DHR is displayed in Fig. 11(f), compared with the label displayed in Fig. 11(a).

To generate noisy fault attributes, we generate a 3-D random field $r(x, y, z)$ based on a normal distribution, smooth it by a 3-D Gaussian filter with a standard deviation of four gridpoints along all directions, and linearly rescale it to the value range of [0, 1]. The resulting smoothed random field, say $r'(x, y, z)$, is set to 0 where $r' < 0.4$, and to 1 otherwise. By multiplying this randomly binary 0–1 mask $r'(x, y, z)$ with the fault attribute labels, we obtain noisy fault attributes with 'cheese holes' on the fault surfaces. Additionally, we

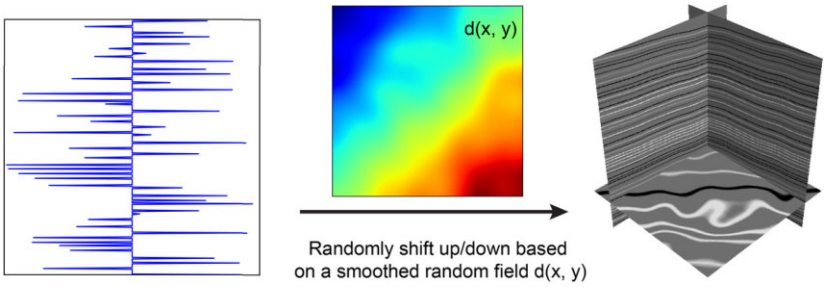

Figure B2. The procedure for generating unfaulted reflectivity volume.

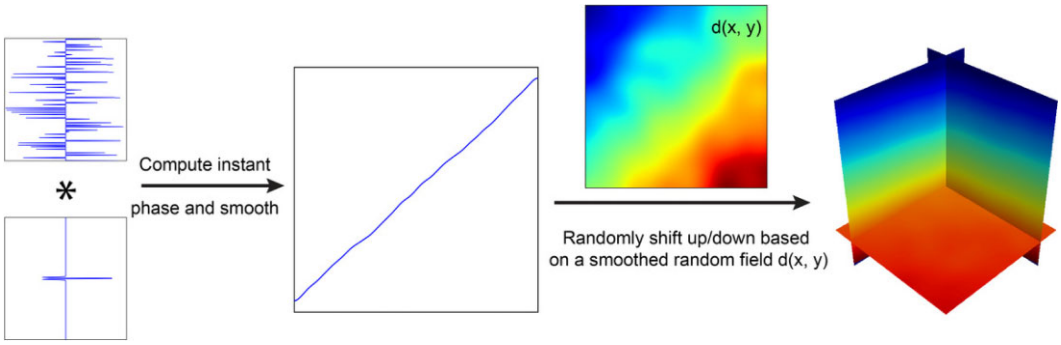

Figure B3. The procedure for generating unfaulted RGT volume. The symbol '$*$' represents convolution.

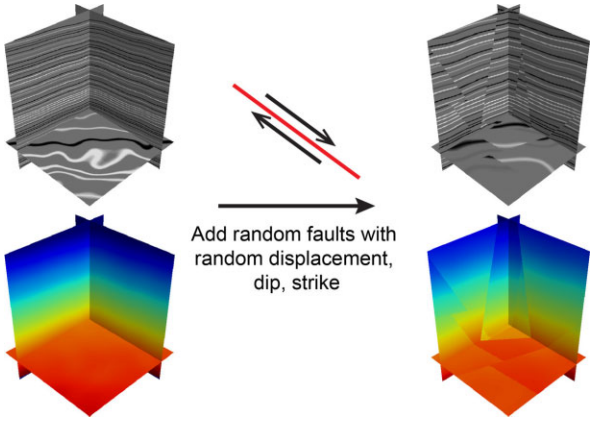

Figure B4. The procedure for faulting the reflectivity and RGT volumes.

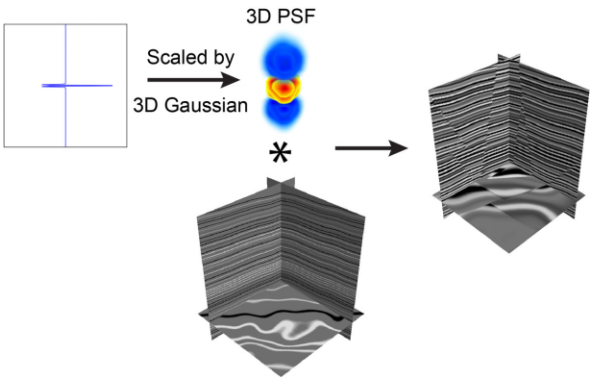

Figure B5. The procedure for generating the noise-free image.

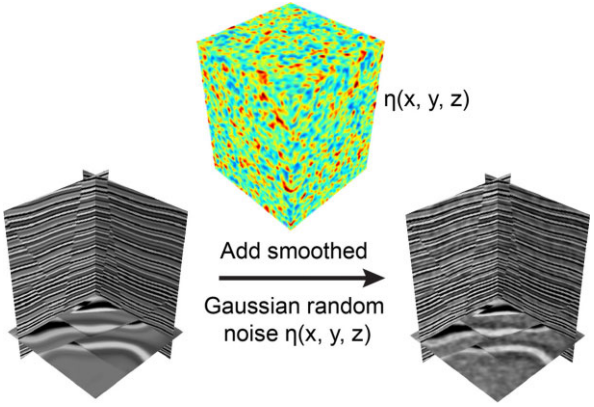

Figure B6. The procedure for generating the noisy image.

generate a separate set of fault attributes, and create masked fault attribute volumes by multiplying them with another randomly binary 0–1 mask r'', where we set $r'' = 0$ where $r'' < 0.65$ and $r'' = 1$ otherwise. The results are random-scatterer-like small faults. We add these randomly scattered faults to the 'cheese-hole' fault attribute volume. With these two steps, we obtain noisy fault attribute volumes that contain both cheese-hole artefacts and randomly scattered small faults. The procedure is illustrated in Fig. B7.

To generate a noisy RGT, we create a 3-D random field $r(x, y, z)$, smooth it by a 3-D Gaussian filter with a standard deviation of five samples along all directions, subtract its mean, normalize it with its standard deviation add a constant factor of one, and multiply this field with the label RGT of MTI-Net. The resulting noisy RGT volume thus contains random fluctuations. The procedure is illustrated in Fig. B8.

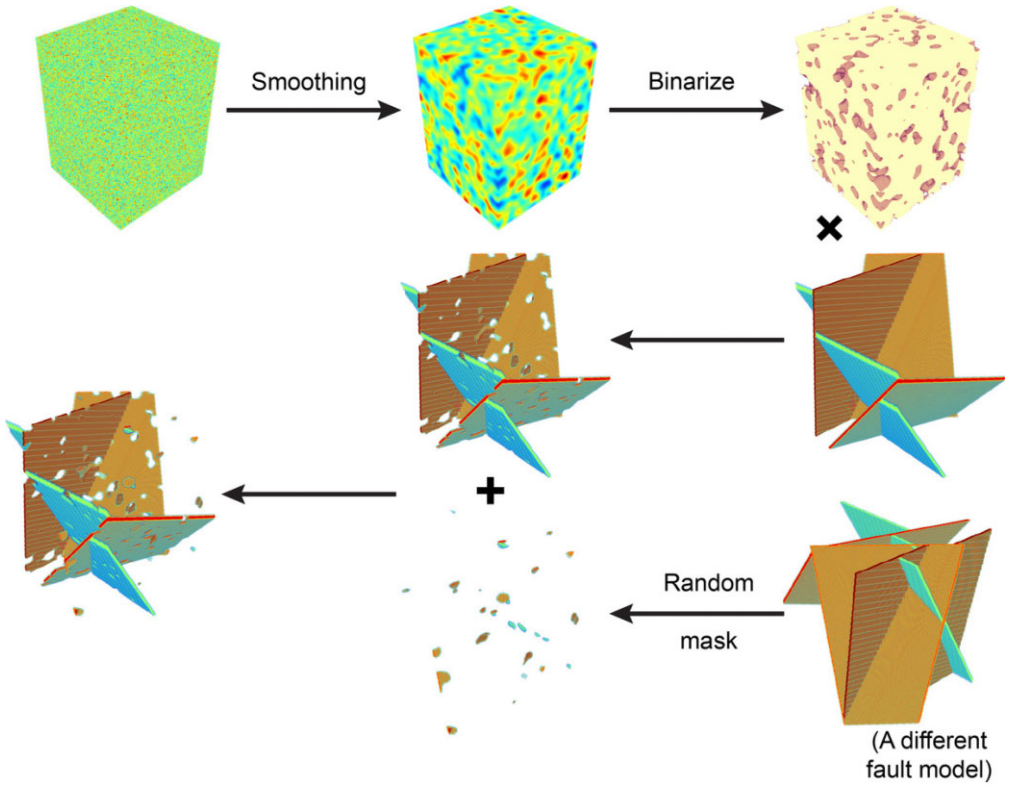

Figure B7. The procedure for generating a noisy fault model (and fault attributes) used as the input when training MTR-Net.

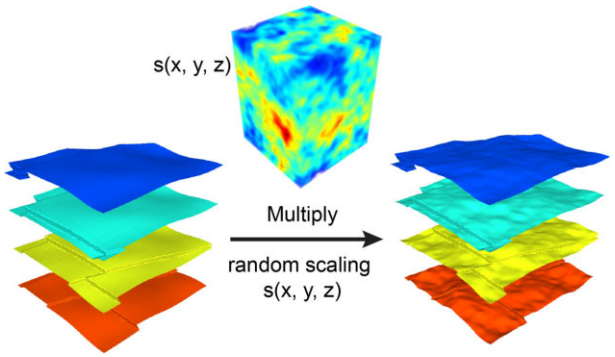

Figure B8. The procedure for generating a noisy RGT volume used as the input when training MTR-Net.