Simulation-based Inference of Reionization Parameters from 3D Tomographic 21 cm Light-cone Images

Xiaosheng Zhao, Yi Mao, Cheng Cheng, Benjamin D. Wandelt

The Astrophysical Journal, 926:151 (19pp), 2022 February 20 https://doi.org/10.3847/1538-4357/ac457d

Simulation-based Inference of Reionization Parameters from 3D Tomographic 21 cm Light-cone Images

Xiaosheng Zhao[1], Yi Mao[1], Cheng Cheng[2], and Benjamin D. Wandelt[3,4,5]
[1] Department of Astronomy, Tsinghua University, Beijing 100084, People's Republic of China; ymao@tsinghua.edu.cn
[2] School of Chemistry and Physics, University of KwaZulu-Natal, Westville Campus, Durban, 4000, South Africa
[3] Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris (IAP), 98 bis bd Arago, F-75014 Paris, France
[4] Sorbonne Université, Institut Lagrange de Paris (ILP), 98 bis bd Arago, F-75014 Paris, France
[5] Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, New York, NY 10010, USA
Received 2021 May 7; revised 2021 December 8; accepted 2021 December 19; published 2022 February 21

Abstract

Tomographic three-dimensional 21 cm images from the epoch of reionization contain a wealth of information about the reionization of the intergalactic medium by astrophysical sources. Conventional power spectrum analysis cannot exploit the full information in the 21 cm data because the 21 cm signal is highly non-Gaussian due to reionization patchiness. We perform a Bayesian inference of the reionization parameters where the likelihood is implicitly defined through forward simulations using density estimation likelihood-free inference (DELFI). We adopt a trained 3D convolutional neural network (CNN) to compress the 3D image data into informative summaries (DELFI-3D CNN). We show that this method recovers accurate posterior distributions for the reionization parameters. Our approach outperforms earlier analysis based on two-dimensional 21 cm images. In contrast, a Monte Carlo Markov Chain analysis of the 3D light-cone-based 21 cm power spectrum alone and using a standard explicit likelihood approximation results in less accurate credible parameter regions than inferred by the DELFI-3D CNN, both in terms of the location and shape of the contours. Our proof-of-concept study implies that the DELFI-3D CNN can effectively exploit more information in the 3D 21 cm images than a 2D CNN or power spectrum analysis. This technique can be readily extended to include realistic effects and is therefore a promising approach for the scientific interpretation of future 21 cm observation data.

Unified Astronomy Thesaurus concepts: Reionization (1383); H I line emission (690); Convolutional neural networks (1938)

1. Introduction

The epoch of reionization (EoR), which marks the formation of the first luminous objects, is one of the milestones in the evolution of our universe. During the EoR, the neutral hydrogen (H I) gas in the intergalactic medium (IGM) is heated and ionized by ultraviolet and X-ray photons from the first luminous objects (see, e.g., Dayal & Ferrara 2018; Hutter et al. 2021). While the observations of high-redshift quasar spectra (e.g., Fan et al. 2006; Becker et al. 2015; McGreer et al. 2015) and the electron scattering optical depth to the cosmic microwave background (CMB; Planck Collaboration et al. 2020) have placed robust observational constraints (Bouwens et al. 2015; Robertson et al. 2015; Finkelstein et al. 2019) on the EoR, the 21 cm line associated with the spin-flip transition of H I atoms will be the most promising probe to cosmic reionization, because, in principle, the tomographic 21 cm survey contains the full three-dimensional (3D) information of H I gas in the EoR, thereby directly revealing the astrophysical processes regarding how H I gas in the IGM was heated and reionized by those ionizing sources.

The next decades will be a golden age for 21 cm observations with a number of ongoing and upcoming experiments. Current interferometric arrays, including the Precision Array for Probing the Epoch of Reionization (PAPER; Parsons et al. 2010), the Murchison Wide field Array (MWA; Tingay et al. 2013), the LOw Frequency Array (LOFAR; van Haarlem et al. 2013), and the Giant Metrewave Radio Telescope (GMRT; Intema et al. 2017), have first attempted to put upper limits on the 21 cm power spectrum from the EoR (Paciga et al. 2013; Pober et al. 2015; Mertens et al. 2020; Trott et al. 2020). In the near future, the Hydrogen Epoch of Reionization Array (HERA; DeBoer et al. 2017) and the Square Kilometre Array (SKA; Mellema et al. 2013) promise to measure the 21 cm power spectrum from the EoR for the first time. More importantly, it is very likely that the SKA will have enough sensitivity to make 3D maps of the 21 cm signal.

Unlike the CMB, the 21 cm signal is highly non-Gaussian, because patchy, bubble-like structures of ionized hydrogen (H II) regions are produced surrounding the ionizing sources. Thus, there is potentially a wealth of information in the 21 cm signal that is not contained in the 21 cm power spectrum, a two-point statistics of 21 cm brightness temperature fluctuations that is traditionally well studied in the literature. It is therefore essential to develop new methods that maximally exploit the full information in the 3D 21 cm images obtained by the SKA. Toward this goal, conventionally, new summary statistics that can only be measured with imaging have been proposed. These include the three-point correlation function (Hoffmann et al. 2019; Jennings et al. 2020), bispectrum (Yoshiura et al. 2015; Shimabukuro et al. 2016, 2017; Majumdar et al. 2018, 2020; Hutter et al. 2020; Saxena et al. 2020; Kamran et al. 2021), one-point statistics (Harker et al. 2009; Shimabukuro et al. 2015; Gorce et al. 2021), topological quantities such as the Minkowski functionals (Gleser et al. 2006; Chen et al. 2019; Kapahtia et al. 2021) and Betti numbers (Giri & Mellema 2021), the cross correlation between the 21 cm line and other

probes, such as the CO line (Gong et al. 2011; Lidz et al. 2011), the C II line (Gong et al. 2012; Beane & Lidz 2018), the kinetic Sunyaev–Zel'dovich (kSZ) effect (Ma et al. 2018; La Plante et al. 2020), and novel techniques such as the antisymmetric cross correlation between the 21 cm line and CO line (Zhou et al. 2021). Since those summary statistics are fully determined by the parameters in the reionization models (hereafter "reionization parameters"), in principle, Monte Carlo Markov Chain (MCMC) methods can be employed to constrain the reionization parameters from measurements of those statistics with futuristic 21 cm experiments (see, e.g., Watkinson et al. 2022), just as the MCMC analysis with the 21 cm power spectrum (Greig & Mesinger 2015, 2017, 2018).

Ideally, an approach that could constrain the reionization parameters directly from the full 3D 21 cm images would probably have no loss of information. In principle, this could be done by writing down the non-Gaussian likelihood of the 21 cm images as a function of the reionization parameters, similar to the MCMC approach for analyzing the distribution of galaxies (Jasche & Wandelt 2013). Nevertheless, such an explicit likelihood approach is a major effort and very computationally demanding.

Recently, machine learning has been extensively applied to 21 cm cosmology (Jensen et al. 2016; Kern et al. 2017; Shimabukuro & Semelin 2017; Hassan et al. 2019, 2020; Schmit & Pritchard 2018; Doussot et al. 2019; Gillet et al. 2019, hereafter referred to as G19; Jennings et al. 2019; Li et al. 2019; List & Lewis 2020; Kern & Liu 2021; Villanueva-Domingo & Villaescusa-Navarro 2021). In particular, G19 demonstrated that the information in the two-dimensional (2D) 21 cm image slices can be exploited to constrain the parameters for reionization and cosmic dawn with 2D convolutional neural networks (CNNs). However, they found that their constraint performance was just comparable to the MCMC analysis with the 21 cm power spectrum. This can be explained by the fact that the 2D images contain only a subset of information in the 3D images. Nevertheless, this opens a new window to constrain reionization parameters, and even cosmological parameters (Hassan et al. 2020), using 21 cm images directly with machine-learning techniques. Our paper is motivated to extend the work of G19 and demonstrate the applicability of the 3D CNN technique (Baccouche et al. 2011; Çiçek et al. 2016; Amidi et al. 2018) to the 21 cm 3D images for constraining the reionization parameters.

It is worthwhile noting that most 21 cm cosmology literature using machine learning did not include the posterior inference for recovered parameters (not to be confused with the estimation of recovery errors with respect to the true parameter values). In this regard, Hortúa et al. (2020a, 2020b) applied Bayesian neural networks (BNNs; Gal & Ghahramani 2016; Perreault Levasseur et al. 2017; Hortúa et al. 2020c) to the 21 cm 2D images, giving an estimate of the reionization parameters and a measure of the uncertainties. Ideally, BNNs can infer a posterior distribution over the network weights and output principled expectation values and uncertainties assuming a simple multivariate Gaussian distribution. For this purpose, various approximations were made in the literature to realize BNNs. However, the inconsistency between different approaches is still a problem (Hortúa et al. 2020a, 2020b, 2020c).

Instead, in this paper, we employ the density-estimation likelihood-free inference (DELFI; Bonassi et al. 2011; Fan et al. 2013; Papamakarios & Murray 2016; Kern et al. 2017; Lueckmann et al. 2017; Alsing et al. 2018; Lueckmann et al. 2018; Alsing et al. 2019, hereafter referred to as A19; Ramanah et al. 2020), a more flexible framework than CNNs to give the posterior inference, including both parameter estimation and Bayesian uncertainty estimation. Unlike Bayesian inference based on an explicit likelihood (Greig & Mesinger 2015, 2017, 2018; Park et al. 2019), the so-called likelihood-free inference (LFI) defines the likelihood implicitly through forward simulations. This allows building a sophisticated data model without relying on approximate likelihood assumptions. Instead, DELFI contains various neural density estimators (NDEs) to learn the likelihood as the conditional density distribution of the target data given the parameters from a number of simulated parameter–data pairs. It has been demonstrated to need fewer model simulations to get the high-fidelity posterior distribution than the traditional Approximate Bayesian Computation (ABC; Cameron & Pettitt 2012; Schafer & Freeman 2012; Weyant et al. 2013; Robin et al. 2014; Akeret et al. 2015; Ishida et al. 2015; Lin & Kilbinger 2015; Carassou et al. 2017; Hahn et al. 2017; Davies et al. 2018; Kacprzak et al. 2018) when comparing the convergence speed. In comparison with BNNs, the optimization of an ensemble of networks in DELFI is typically simpler and cheaper than the posterior inference over the large number of weights of a single large network in BNNs. In addition, it is easy for DELFI to average different network architectures by taking advantage of neural ensembles (A19).

In this paper, we first train a 3D CNN, which outputs the predicted values of the reionization parameters from the 21 cm light-cone 3D images. While these estimates have physical meaning, note that from the DELFI point of view, these parameter values predicted from the 3D CNN are data *summaries* of the input 3D images, and the 3D CNN is just a data *compressor*. DELFI itself is a framework of posterior inference to provide both parameter estimation (technically, independent of the 3D CNN prediction) and statistical uncertainty estimation. We then apply a public implementation of DELFI with NDEs, `pydelfi`[6] (A19), to perform the posterior inference. As a demonstration of concept, we simplify our simulations of mock samples in two aspects. First, while both reionization parameters and cosmic dawn astrophysical parameters were constrained in G19, our simulations are restricted to the regime when the 21 cm spin temperature is much larger than the CMB temperature, so the contributions from the cosmic dawn astrophysical parameters are negligible. This assumption holds well after reionization begins. Second, as in G19, our simulations do not include the effects of thermal noise and residual foregrounds. Since the DELFI framework is flexible enough, these effects can be readily taken into account in future work. Our paper improves upon G19 in exploiting the 3D 21 cm images with the 3D CNN and performing a self-consistent posterior inference with `pydelfi`.

The rest of this paper is organized as follows. In Section 2, we introduce our integrated framework of DELFI with the 3D CNN as a data compressor (hereafter "DELFI-3D CNN"). We give the results of the DELFI-3D CNN in Section 3 and make concluding remarks in Section 4. Some technical discussions are left to Appendix A (on the optimization of 3D CNN), Appendix B (on the effect of missing $\boldsymbol{k}_\perp = 0$ mode in the interferometric measurement), Appendix C (on the formalism of NDEs), Appendix D (on the effect of initial conditions),

[6] https://github.com/justinalsing/pydelfi

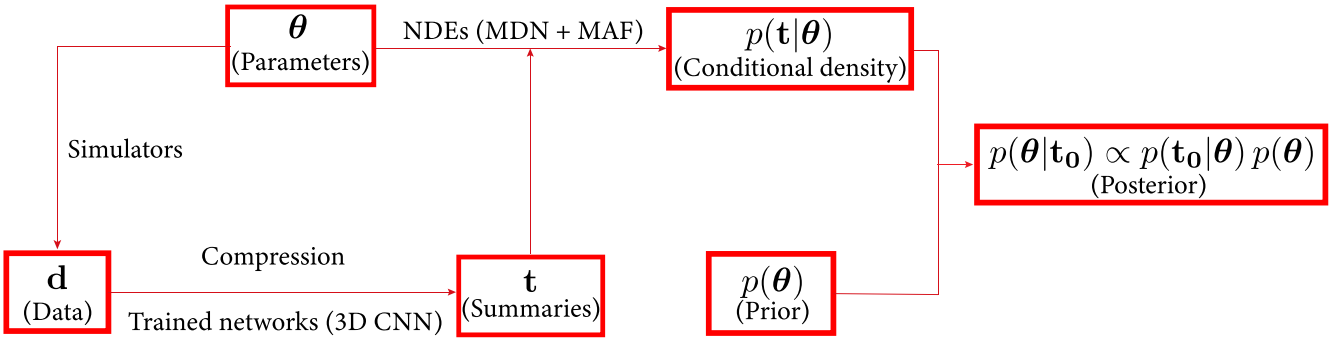

Figure 1. The workflow of the DELFI-3D CNN. The data $\boldsymbol{d}$ are prepared using the simulator (`21cmFAST` herein) with the parameters $\boldsymbol{\theta}$. The data are compressed to the low-dimensional summaries $\boldsymbol{t}$ using some compressors (trained 3D CNN herein). The parameter–summary pairs ($\boldsymbol{\theta}$, $\boldsymbol{t}$) are fed to an ensemble of NDEs that are trained to learn the data likelihood $p(\boldsymbol{t}|\boldsymbol{\theta})$. The posterior distribution is inferred from the learned data likelihood and parameter prior using Bayes' theorem.

Appendix E (on the stabilization of different NDEs), and Appendix F (on the convergence of the `21CMMC` code).

2. DELFI-3D CNN Methodology

DELFI transforms the problem of parameter inference into density estimation with the simulated parameter–data pairs. To implement DELFI, we adopt a flexible and effective approach, with the workflow illustrated in Figure 1—we first learn the conditional density $p(\boldsymbol{t}|\boldsymbol{\theta})$ by maximizing the total data likelihood, where $\boldsymbol{\theta}$ and $\boldsymbol{t}$ are the parameter vector and data summary vector, respectively, and then infer the posterior using Bayes' theorem, $p(\boldsymbol{\theta}|\boldsymbol{t}_0) \propto p(\boldsymbol{t}_0|\boldsymbol{\theta})\, p(\boldsymbol{\theta})$ at any data summary $\boldsymbol{t}_0$ from observed data, where $p(\boldsymbol{\theta})$ is the prior. In view of large-scale data from simulations, we employ the 3D CNN to compress the 21 cm 3D images (the data $\boldsymbol{d}$) into the data summaries $\boldsymbol{t}$ and train DELFI with a large number of parameter–summary pairs ($\boldsymbol{\theta}$, $\boldsymbol{t}$). In this section, we introduce our methodology in detail.

2.1. Data Preparation

In this paper, we use the publicly available code `21cmFAST`[7] (Mesinger & Furlanetto 2007; Mesinger et al. 2011), which can be used to perform seminumerical simulations of reionization, as the simulator to generate the data sets. This code uses the excursion-set approach (Furlanetto et al. 2004b) to identify ionized regions. It quickly generates the fields of density, velocity, ionization field, spin temperature, and 21 cm brightness temperature on a grid.

Our simulations were performed on a cubic box of 100 comoving Mpc on each side, with 66^3 grid cells. The 21 cm brightness temperature at position $\boldsymbol{x}$ relative to the CMB temperature can be written (Furlanetto et al. 2006) as

$$T_{21}(\boldsymbol{x}, z) = \tilde{T}_{21}(z) x_{\mathrm{H\,I}}(\boldsymbol{x})[1 + \delta(\boldsymbol{x})]\left(1 - \frac{T_{\mathrm{CMB}}}{T_S}\right), \qquad (1)$$

where $\tilde{T}_{21}(z) = 27\sqrt{[(1+z)/10](0.15/\Omega_{\mathrm{m}}h^2)}\,(\Omega_{\mathrm{b}}h^2/0.023)$ in units of mK. Here, $x_{\mathrm{H\,I}}(\boldsymbol{x})$ is the neutral fraction, and $\delta(\boldsymbol{x})$ is the matter overdensity at position $\boldsymbol{x}$. We assume the baryon perturbation traces the cold dark matter on large scales, so $\delta_{\rho_{\mathrm{H}}} = \delta$. In this paper, we focus on the limit where the spin temperature is $T_S \gg T_{\mathrm{CMB}}$, which is valid soon after reionization begins. As such, we can neglect the dependence on the spin temperature. Also, for simplicity, we ignore the effect of peculiar velocity, because it only weakly affects the light-cone effect.

We generate a realization of the 21 cm brightness temperature fields using the density and ionized fraction fields from semi-numerical simulations with the code `21cmFAST`, given an initial condition in the density fields. We then interpolate the snapshots at different time to construct the light-cone data cube along the line of sight (LOS) within a comoving distance of a simulation box. To reduce the interpolation error caused by insufficient sampling of snapshots, we output the simulation results at nine different redshifts within the corresponding cosmic time. To avoid the impact of periodic boundary condition, which otherwise results in the repeated structures along the LOS due to the same initial density fields, we concatenate 10 such light-cone boxes, each simulated with different initial conditions in density fields but with the same reionization parameters, together to form a full light-cone data cube of a size of $100 \times 100 \times 1000$ comoving Mpc^3. To mimic the observations from radio interferometers, we subtract from the light-cone field the mean of the 2D slice for each 2D slice perpendicular to the LOS, because radio interferometers cannot measure the mode with $\boldsymbol{k}_\perp = 0$.[8] This forms the mock light-cone data cube of $\delta T_{\mathrm{b}} - \overline{\delta T_{\mathrm{b}}}$ in the redshift range $7.51 \leqslant z \leqslant 11.67$ (see Figure 2).

Our reionization model is parameterized with two parameters as follows:

(1) ζ, the *ionizing efficiency*, $\zeta = f_{\mathrm{esc}} f_* N_\gamma/(1 + \bar{n}_{\mathrm{rec}})$ (Furlanetto et al. 2004a, 2006), which is a combination of several parameters related to ionizing photons. Here, f_{esc} is the fraction of ionizing photons escaping from galaxies into the IGM, f_* is the fraction of baryons locked in stars, N_γ is the number of ionizing photons produced per baryon in stars, and $\bar{n}_{\mathrm{rec}}$ is the mean recombination rate per baryon. In our data set, we vary ζ in the range of $10 \leqslant \zeta \leqslant 250$.

(2) T_{vir}, the *minimum virial temperature of halos that host ionizing sources*. Typically, T_{vir} is about 10^4 K, corresponding to the temperature above which atomic cooling becomes effective. In our data set, we explore the range of $10^4 \leqslant T_{\mathrm{vir}} \leqslant 10^6$ K.

These ranges are consistent with Greig & Mesinger (2018). Other parameters that have less impact on reionization are fixed throughout this paper, e.g., $R_{\mathrm{mfp}} = 15$ Mpc (the mean free path of ionizing photons). Also, cosmological parameters are fixed as $(\Omega_\Lambda,\ \Omega_m,\ \Omega_b,\ \mathrm{n}_s,\ \sigma_8,\ h) = (0.692,\ 0.308,\ 0.0484,\ 0.968,\ 0.815,\ 0.678)$ (Planck Collaboration et al. 2016).

[7] https://github.com/andreimesinger/21cmFAST

[8] Note that G19 did not implement this step of subtracting the mean of the 2D slice from the 21 cm signal. The effect of missing the $\boldsymbol{k}_\perp = 0$ mode in the interferometer measurement is discussed in Appendix B.

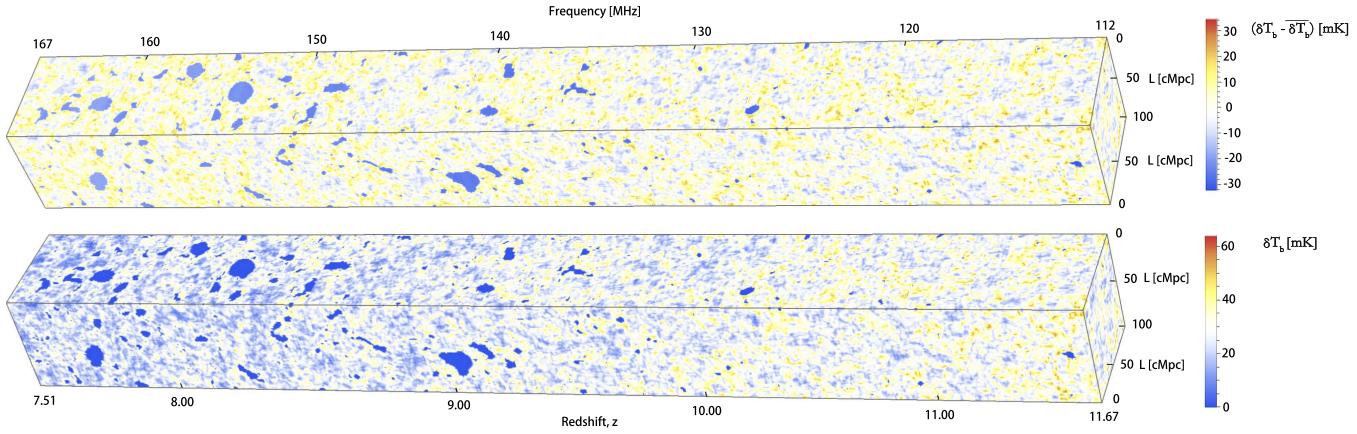

Figure 2. An illustration of the light-cone data cube of 3D 21 cm images δT_b (bottom) in the redshift range $z = 7.51$–11.67. To mimic the observations from radio interferometers, we subtract from the light-cone field the mean of the 2D slice for each 2D slice perpendicular to the LOS and form the observed 3D 21 cm images (top), $\delta T_b - \overline{\delta T_b}$.

For each given set of reionization parameters, we generate a mock light-cone data cube of 3D 21 cm images, or a sample. We use the Latin Hypercube Sampling (McKay et al. 1979) to scan the EoR parameter space. For the 3D CNN, we generate 10,000 samples, in which 8000 samples are used for training the 3D CNN, 1000 for validation, and 1000 for testing the 3D CNN. After the 3D CNN is trained, its network weights are fixed. Then we generate 10,000 new samples which are compressed into parameter–data summaries pairs by the trained 3D CNN and then provided to DELFI for training the density estimators (with 8100 samples), validation (with 900 samples), and testing the DELFI-3D CNN (with 1000 samples). The initial conditions for all realizations were independently generated by sampling spatially correlated Gaussian random fields with the power spectrum given by linear theory.[9]

2.2. 3D CNN

We train a 3D CNN to get an estimator of the reionization parameters from the 3D 21 cm light-cone data cube. From the DELFI perspective, the trained 3D CNN is a compressor that compresses the 3D 21 cm images (the data $\boldsymbol{d}$) into the estimates of reionization parameters (the summaries $\boldsymbol{t}$). In this subsection, we describe in detail the structure of a typical 3D CNN, the step-by-step process of feature extraction in 3D CNN, the training strategy, and some practical aspects of hyperparameter tuning.

2.2.1. Structure of 3D CNN

Basically, a 3D CNN can be set up by replacing the 2D operations with 3D counterparts, including 3D Convolution (Conv3D), 3D MaxPooling, 3D ZeroPadding, and 3D Dropout. We illustrate the main structure in the 3D CNN in Figure 3. Interested readers are referred to G19 for the 2D CNN that was applied to astrophysical parameter estimation.

A network can be built by stacking many layers together. When the input image goes through the convolutional operations, the image features are extracted by some 3D kernels each with certain size. Each kernel sweeps through the entire image with a specific stride size in a specific order. After that, one corresponding feature map is generated. A ZeroPadding operation before the Convolution operation can be used to offset the shape size decrease in the output image from the former layer. A BatchNormalization operation (Ioffe & Szegedy 2015; Santurkar et al. 2018) can follow a Convolution one to enable faster and more stable training. The batch normalization first normalizes the input to have the expectation value of 0 and the variance of 1, and then introduces a linear transform to the normalized values to ensure identity transform (Ioffe & Szegedy 2015). It is usually applied to minibatches, which are drawn from the whole training data. Nonlinear functions like rectified linear units (ReLU; Dahl et al. 2013), $f(x) = \max(0, x)$, are applied to enable the network to create complex mappings between the inputs and outputs. After the Convolution layer, a Pooling operation, which serves as a downsampling strategy, can reduce the number of parameters and learn position invariant features. In the fully connected layer, neurons have full connections to all activations in the former layer. The Dropout operation (Baldi & Sadowski 2013; Srivastava et al. 2014) can be applied to tackle potential overfitting of the network by randomly dropping units from the neural network during training, preventing units from coadapting too much. Finally, the output of the network can be either class labels for classification problems or continuous variables for regression problems. The parameter estimation is a typical example of a regression problem.

2.2.2. Network Training

For regression problems, loss functions like the mean absolute error and mean squared error aim to penalize the deviation between the true and predicted values. Our objective is to attain a loss minimum where the loss function reaches the smallest possible value, which may be facilitated by using a proper optimizer like `RMSprop` (Tieleman & Hinton 2012). The key ingredient for an optimizer is the recipe for an appropriate learning rate: too large a learning rate often leads to local minima, but convergence is hard to achieve if the learning rate is too small. In this paper, we choose a variable learning rate that decreases by a factor of 10 once the learning begins to stagnate.

Generally, a validation data set that is independent of the training and test data sets is used during the training process, for two purposes—to choose the optimal model on the

[9] The effect of initial conditions on the parameter estimation is discussed in Appendix D.

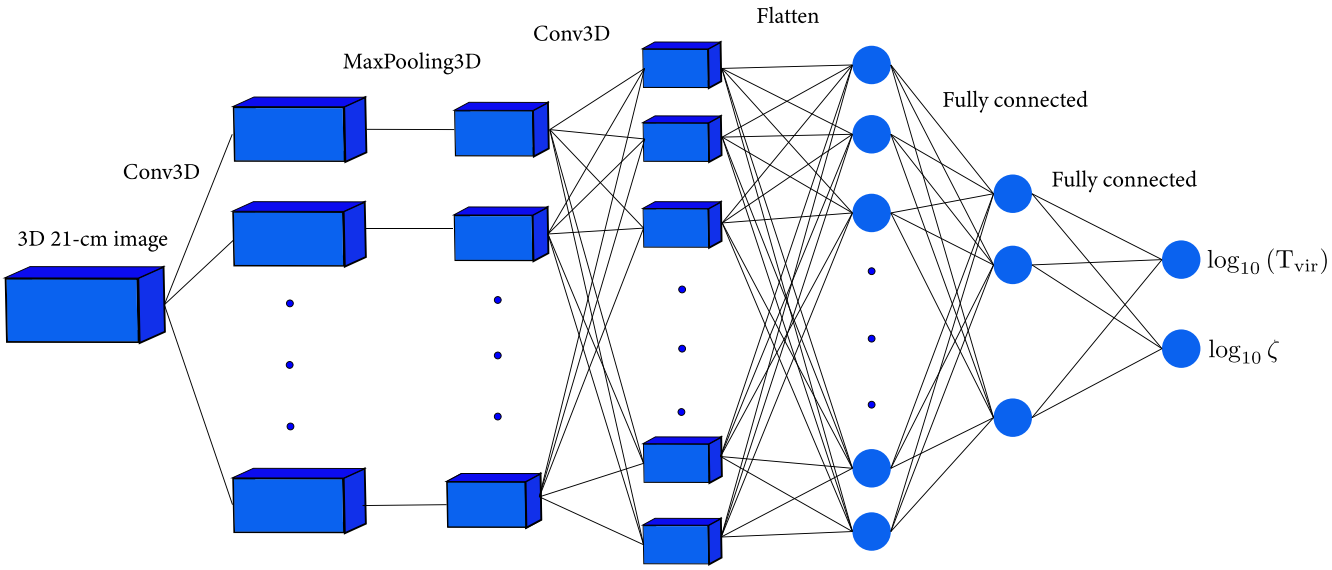

Figure 3. An illustration of the main structure in the 3D CNN. The input set is the 3D 21 cm images, and the output is a set of two reionization parameters, $\log_{10}(T_{\rm vir})$ and $\log_{10}(\zeta)$. Here we show the main 3D Convolution (Conv3D), Maxpooling3D, Flatten, and fully connected layers.

validation data set and avoid overfitting while training. Overfitting can be recognized if errors on the validation data set increase. A strategy for faster convergence to an optimal model is to set early stopping, which ends the training when errors on the validation data set stop decreasing. Once that happens, we can choose a new set of hyperparameters for training. In view of the GPU memory limitation, we can split the training data set into "minibatches". After processing one minibatch of training data, the network updates the network weights through backpropagation, which are the actual parameters trained throughout the network. Note that inefficient computation occurs (due to noisy stochastic gradients) when the size of the minibatch is too small, so the minibatch size serves as a trade-off between the memory limitation and computational efficiency. We choose a minibatch size of 32 in this paper.

We adopt a single 3D CNN model that is however flexible enough. The network setup includes one Convolutional layer with filter size of $5 \times 5 \times 5$, one MaxPooling layer, two blocks (each containing one ZeroPadding layer, one Convolutional layer, one MaxPooling layer, and one 3D Dropout layer), a Flatten layer, one fully connected layer with 64 neurons, one 1D Dropout layer, and two successive fully connected layers with 16 and 4 neurons, respectively. The dropout rate for all kinds of Dropout layers is 0.4. The total number of parameters to be trained is about 2.56 million. The layer types and their output shapes are listed in Table 1. We use the Keras functional API[10] to set up the CNN model.

Due to the large dynamical range of reionization parameters, we choose to output these parameters in the logarithmic scale, in the default range $(4 \leqslant \log_{10}(T_{\rm vir}/{\rm K}) \leqslant 6,\ 1 \leqslant \log_{10}(\zeta) \leqslant 2.398)$. We also optimize the network with the output weights chosen to be 0.8 and 1.0 for $\log(T_{\rm vir})$ and $\log(\zeta)$, respectively, in order to balance the recovery performance between these two parameters, because the 21 cm signals depend on them with slightly different sensitivity (see the 2D case in G19). The recovery performance is roughly stabilized among different network setups, which we discuss in detail in Appendix A. The aforementioned setups, which we adopt throughout this paper unless noted otherwise, result from the optimization of the recovery performance. In this paper, we use 64 GB memory for loading and training data, and one NVIDIA GeForce GTX 1080 Ti GPU card with 11 GB RAM memory for computational speedup. It takes 6 minutes for each epoch of training, and a typical model training is finished within 70 epochs. After training, it takes 18 ms for the trained 3D CNN to process a test sample.

[10] https://keras.io/getting-started/functional-api-guide/

Table 1
The Stacked Layers with their Output Shapes in the 3D CNN

Step	Layer type	Output shape
1	Input	$66 \times 66 \times 660$
2	$5 \times 5 \times 5$ Conv3D	$31 \times 31 \times 328$
3	BatchNormalization+ReLU	$31 \times 31 \times 328$
4	$2 \times 2 \times 2$ MaxPooling3D	$15 \times 15 \times 164$
5	ZeroPadding3D	$17 \times 17 \times 166$
6	$5 \times 5 \times 5$ Conv3D	$13 \times 13 \times 162$
7	BatchNormalization+ReLU	$13 \times 13 \times 162$
8	$2 \times 2 \times 2$ MaxPooling3D	$6 \times 6 \times 81$
9	SpatialDropout3D(40%)	$6 \times 6 \times 81$
10	ZeroPadding3D	$8 \times 8 \times 83$
11	$5 \times 5 \times 5$ Conv3D	$4 \times 4 \times 79$
12	BatchNormalization+ReLU	$4 \times 4 \times 79$
13	$2 \times 2 \times 2$ MaxPooling3D	$2 \times 2 \times 39$
14	SpatialDropout3D(40%)	$2 \times 2 \times 39$
15	Flatten	19968
16	Fully connected	64
17	BatchNormalization+ReLU	64
18	Dropout(40%)	64
19	Fully connected	16
20	BatchNormalization+ReLU	16
21	Fully connected	4
22	BatchNormalization+ReLU	4
23	Fully connected Left	1
24	Fully connected Right	1

2.3. Neural Density Estimators (NDEs)

NDEs can model the probability distribution (or density) conditioned on the parameters $p(\boldsymbol{t}|\boldsymbol{\theta})$. They are powerful tools

for DELFI. Two implementations of NDEs have been proven successful—mixture density networks (MDNs; Bishop 1994) and masked autoregressive flows (MAFs; Papamakarios et al. 2017). We give a brief summary in this subsection and leave the technical details to Appendix C, based on the work of Bishop (1994), Germain et al. (2015), Papamakarios et al. (2017), Papamakarios et al. (2021), and (A19).

2.3.1. Mixture Density Networks (MDNs)

MDNs combine typical neural networks with a mixture density model such as Gaussian density. It can represent more complexity of the conditional density when more mixture components are contained. With enough Gaussian components, in principle, MDNs can model the underlying non-Gaussian properties in the conditional data density.

2.3.2. Masked Autoregressive Flows (MAFs)

The word "flows" here refers to the normalizing flows (Papamakarios et al. 2021), which transform a simple base density with a series of transformations to a richer distribution, in analogy to a fluid flowing through a set of tubes. The base density is often taken to be a multivariate normal, so the inverse transformation from any distribution back to the base density is called "normalizing".

The transformation is required to be invertible and both the forward transformation and inverse transformation should be differentiable. These properties also ensure that the transformations are composible, which increases the actual expressive power of flow-based models.

With a flow-based model, one can either draw samples from the model with the base density and the forward transformation, or evaluate the model's density with the inverse transform and its Jacobin determinants and the base density. These two options can be chosen for specific needs. In this paper, we need to evaluate the model's density so we choose the latter one.

The autoregressive model can be interpreted as a normalizing flow. For the autoregressive model, the density of vector $\boldsymbol{t}$ can be expressed as a product of one-dimensional (1D) density by the chain rules, where the autoregressive property means that the density of the component t_i only depends on the previous $i-1$ components. The masked autoencoders for density estimation (MADE) provide an efficient way to implement the autoregressive property with a modified fully connected autoencoder. Consider a Gaussian distribution as the individual 1D density, by learning the mean and variance of each density and using the exponential activation function to ensure the positivity, the MADE can be interpreted as the inverse transformation from the vector $\boldsymbol{t}$ back to a random vector following a standard normal distribution. The autoregressive property makes the Jacobin determinants tractable, which is feasible for density estimation.

In practice, we stack multiple MADEs to form the normalizing flows. The ordering of input data components to construct the autoregressive density can be varied for each MADE in order to increase the flexibility. In this paper, the MAFs are conditioned on parameter $\boldsymbol{\theta}$. This can be easily implemented by constructing the chain rule of densities conditioned on $\boldsymbol{\theta}$.

Training the NDEs (including both MAFs and MDNs) is basically fitting the density estimators $p(\boldsymbol{t}|\boldsymbol{\theta};\boldsymbol{w})$ with dependence on the network weights $\boldsymbol{w}$ to a target density $p^{\star}(\boldsymbol{t}|\boldsymbol{\theta})$ by minimizing some divergence, e.g., the Kullback–Leibler divergence, between them.

2.3.3. NDE Setup

Both MDNs and MAFs are included in the `pydelfi` package. Training an ensemble of NDEs can help to get robust results on small training sets, as well as avoiding the risk of overfitting (A19). We construct herein the density estimator from an ensemble of four MDNs (each with 1, 4, 6, and 8 Gaussian components, respectively), and a MAF (with 8 MADEs). We checked and confirmed that other ensembles of networks give similar results (see Appendix E).

We train the NDEs with the batch size of 100, epochs of 500, and early stop patience of 20 (which means that the training is finished when there is no loss decreasing over 20 epochs).

2.4. Other Setups of DELFI

For the input of DELFI, the trained 3D CNN with fixed weights compresses the 21 cm 3D images into the 2D summaries (corresponding to the 2D reionization parameter space).

The choice of prior in the parameters is flexible. If active learning is taken, e.g., in (A19), the prior is given by the learning process. In our paper, we choose a flat prior $p(\boldsymbol{\theta})$ over the default region in the parameter space and produce the training set using parameters θ sampled from it before running DELFI.

To sample the posteriors, we run an MCMC on the Bayesian posterior output by DELFI. We run 100 walkers for 3000 steps and drop the first 500 steps as "burn-in," because the integrated autocorrelation time is estimated to be around 30. We have tested the results of posterior inference with a larger number (up to 100,000) of total steps and find the convergence for $\geqslant 3000$ steps.

3. Results

3.1. Parameter Estimation with 3D CNN

After the 3D CNN is trained with 8000 training samples, it is tested against 1000 test samples. For each test sample, we compare the output reionization parameter set predicted from the 3D CNN with the true parameter value of this sample. The overall recovery performance can be evaluated by the coefficient of determination R^2, defined as

$$R^2 = 1 - \frac{\sum (y_{\rm pred} - y_{\rm true})^2}{\sum (y_{\rm true} - \bar{y}_{\rm true})^2}, \tag{2}$$

where $y_{\rm true}$ and $y_{\rm pred}$ are the true and predicted parameter values for a variable y in a sample, respectively, and the summation for this variable is over all test samples. $\bar{y}_{\rm true}$ is the average of the true value in all test samples. In general, a score of R^2 closer to unity indicates a better overall recovery performance of this variable. For our trained 3D CNN, $R^2 = 0.993$ and 0.983 for $\log(T_{\rm vir})$ and $\log(\zeta)$, respectively. (G19 also used the R^2 value to evaluate the output performance. However, they recover ζ, instead of $\log(\zeta)$. Our result for ζ is $R^2 = 0.953$.)

We further test the reionization parameter recovery performance by comparing the predicted parameter values from the trained 3D CNN with the true values in Figure 4. Our results show that the recovery is mostly near the perfect, zero-error

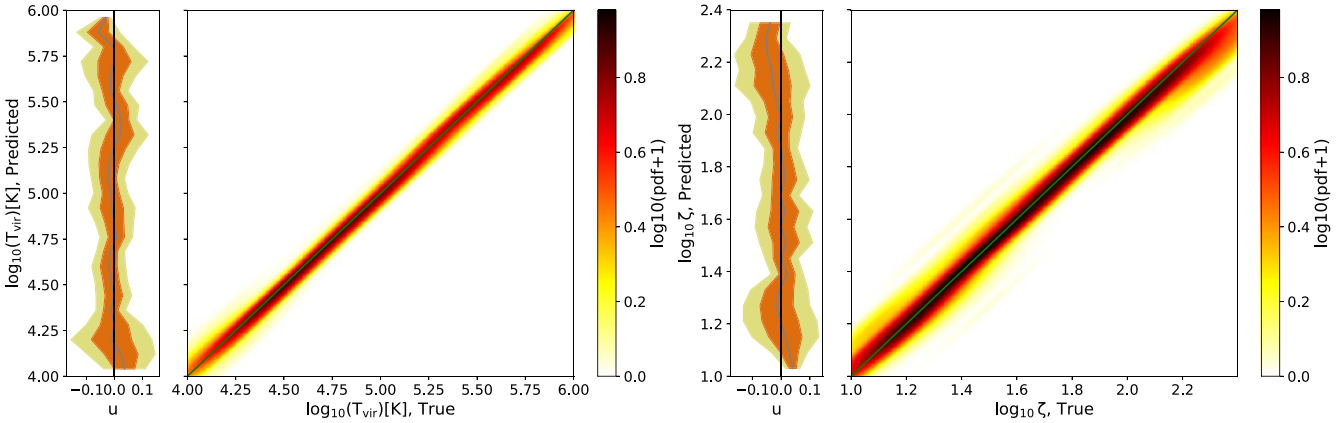

Figure 4. Reionization parameter recovery performance with 3D CNN using 1000 test samples. Shown is the predicted value $y_{\rm pred}$ vs. the true value $y_{\rm true}$ of each parameter $y = \log_{10}(T_{\rm vir})$ (left) and $\log_{10}(\zeta)$ (right), respectively, with the color representing $\log_{10}(1 + {\rm pdf})$ where pdf is the probability density function of the sampled data points. The green diagonal line indicates the perfect (zero-error) recovery. The side histograms show the distribution of residuals $u = y_{\rm pred} - y_{\rm true}$ at a given predicted value, with the mean (gray line) and 68% (orange region) and 95% (yellow region) probability intervals.

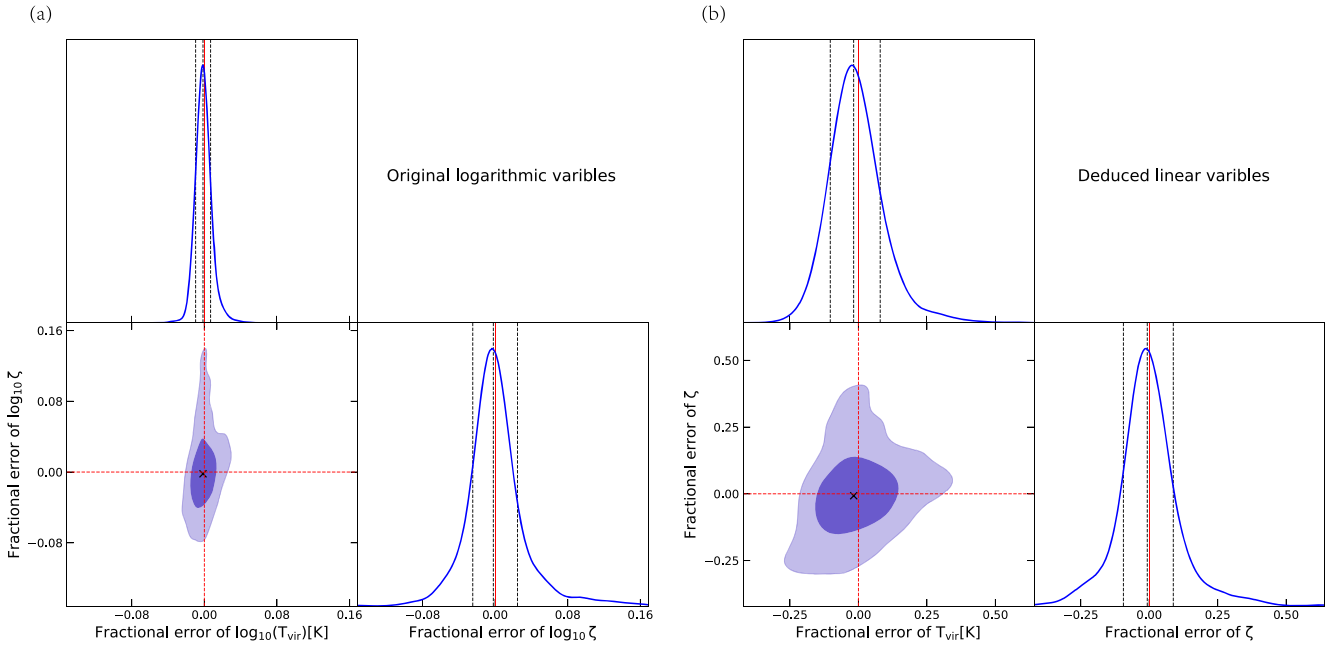

Figure 5. Reionization parameter recovery performance with 3D CNN using 1000 test samples. Shown are the 2D joint probability density distribution of the accuracy of the estimated parameters, i.e., the fractional error $\epsilon = (y_{\rm pred} - y_{\rm true})/y_{\rm true}$, where y represents either the parameters directly predicted from the 3D CNN, $\log(T_{\rm vir})$ and $\log(\zeta)$ (left), or the deduced parameters, $T_{\rm vir}$ and ζ (right). In the 1D marginalized distribution of each panel, the quantiles of 0.16, 0.5, and 0.84 are marked with black dash lines, while the red solid line indicates the perfect (zero-error) recovery. In the 2D joint probability density distribution of each panel, we show the 68% (dark purple region) and 95% (light purple region) probability regions, the median (black cross), and the perfect (zero-error) recovery (red dash lines).

recovery (diagonal line), with reasonably small scatter (which will be quantified by the fractional error below). The side histograms show the distribution $p(u|y_{\rm pred})$ at a given predicted value $y_{\rm pred}$, where the residual $u = y_{\rm pred} - y_{\rm true}$. In the 68% probability interval, $u \lesssim 0.05$, which means that the fractional error in $T_{\rm vir}$ or ζ is about $0.05/\lg(e) \approx 12\%$, as the output parameter y is in the logarithmic scale of base 10. We also note in Figure 4 that near both upper and lower limits of the parameter range, the scatters from the perfect recovery become larger, which means worse recovery accuracy. This is probably due to boundary effects in the parameter space, i.e., the networks cannot take the data outside the boundaries. This could be avoided in practice by choosing the range of the parameter space wide enough around the most likely values indicated by a trial analysis.

The recovery performance test can be further made by plotting in Figure 5 the 2D joint probability density distribution of the fractional errors, $\epsilon = (y_{\rm pred} - y_{\rm true})/y_{\rm true}$, where y represents either the parameters directly predicted by the CNN (in the logarithmic scale), $\log(T_{\rm vir})$ and $\log(\zeta)$, or the deduced parameter (in the linear scale), $T_{\rm vir}$ and ζ. We employ `GetDist` (Lewis 2019) to make the contour plots throughout this paper. Within the 68% probability regions in the joint probability density distribution, the fractional error with respect to the true value is 0.01 for $\log_{10}(T_{\rm vir})$ and 0.03 for $\log_{10}(\zeta)$, or 0.09 for $T_{\rm vir}$ and 0.12 for ζ.

The above tests demonstrate that the 3D CNN can recover the reionization parameters within reasonable accuracy ($\lesssim 12\%$ error), albeit not perfectly. Between the two parameters, $T_{\rm vir}$ is recovered with better accuracy than ζ, even if the output weight

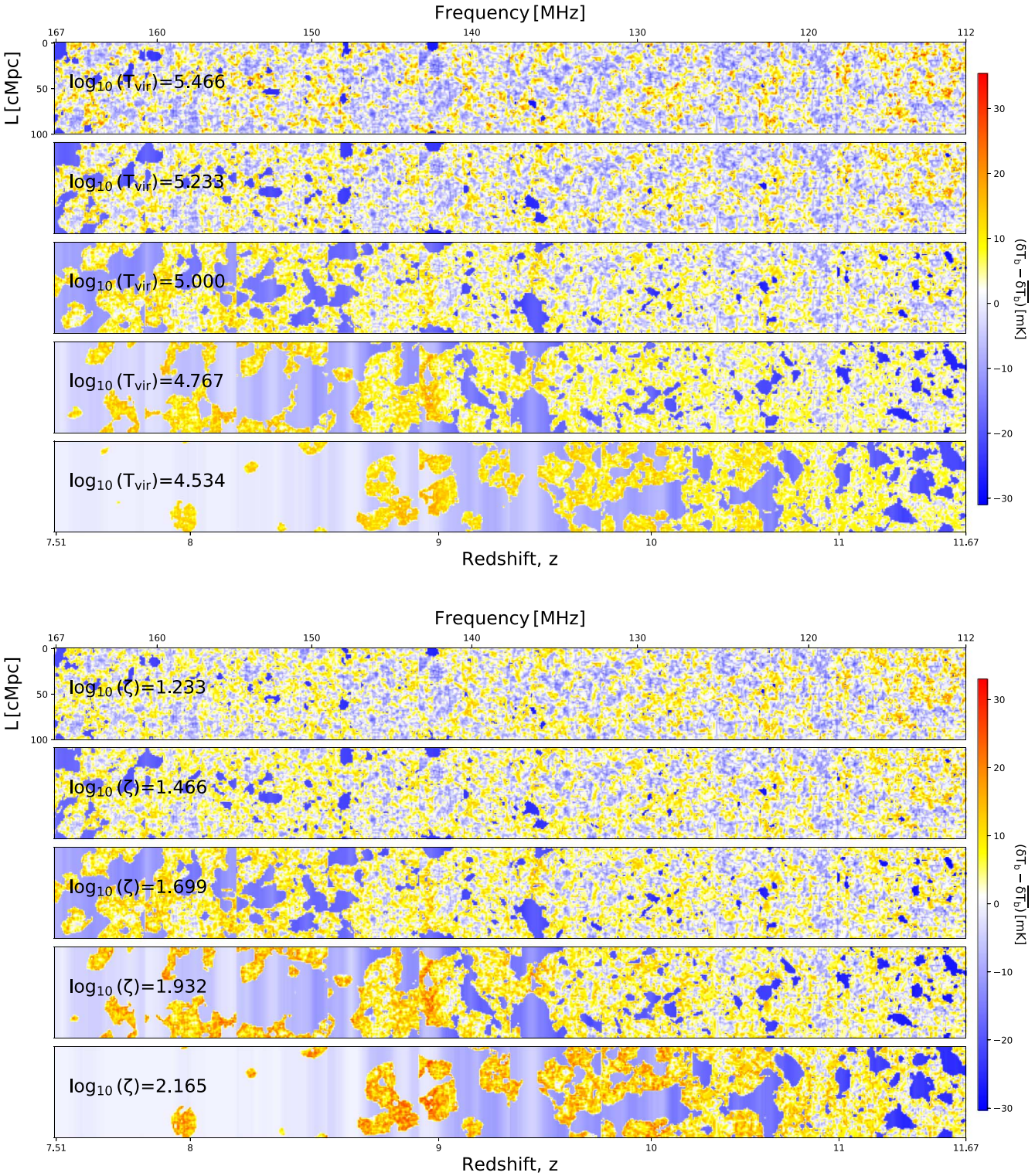

Figure 6. An illustration of the impact of varying parameters on the signal. (Top) Light-cone 21 cm images with fixed $\log_{10}(\zeta) = 1.699$ and varying $\log_{10}(T_{\rm vir}) = 5.466, 5.233, 5.000, 4.767, 4.534$, respectively. (Bottom) Light-cone 21 cm images with fixed $\log_{10}(T_{\rm vir}) = 5.000$ and varying $\log_{10}(\zeta) = 1.233, 1.466, 1.699, 1.932, 2.165$, respectively. The middle light cones in both top and bottom panels are the same and used as the benchmark for comparison.

for $\log(T_{\rm vir})$ (0.8) is smaller than that for $\log(\zeta)$ (1.0). This implies that the 21 cm images are more sensitive to $T_{\rm vir}$ than ζ. This may be explained in Figure 6 by showing the light-cone 21 cm images with varying reionization parameters $\log_{10}(T_{\rm vir})$ (top panel) and $\log_{10}(\zeta)$ (bottom panel), respectively. For the sake of fair comparison, the step size for both parameters are the same, i.e., $\Delta\log_{10}(T_{\rm vir}) = \Delta\log_{10}(\zeta) = 0.233$. Both decreasing $T_{\rm vir}$ and increasing ζ can speed up cosmic reionization. While their impacts on the images are very similar, we find that changing $\log_{10}(T_{\rm vir})$ affects the light cone to a slightly larger extent than changing $\log_{10}(\zeta)$, when one carefully compares the morphology of H II bubbles. More quantitatively, the mean neutral fraction of

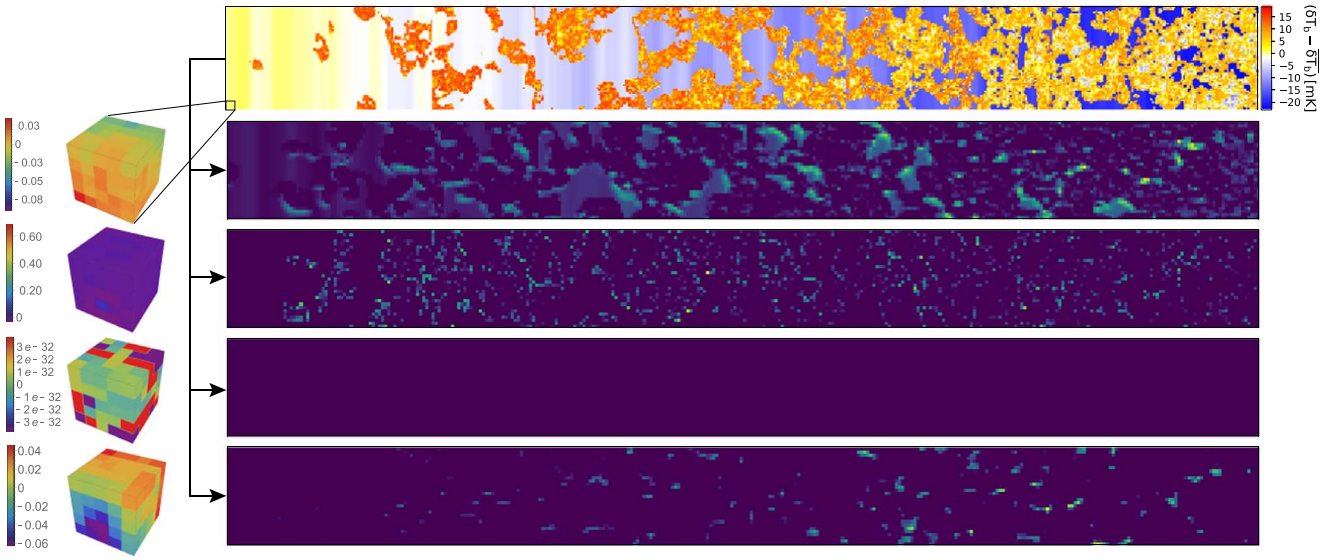

Figure 7. An illustration of kernels in the 3D CNN. (Top) Input light-cone 21 cm images in $66 \times 66 \times 660$ pixels with parameters $\log_{10}(T_{\rm vir}) = 4.101$ and $\log_{10}(\zeta) = 1.389$. (Second top to bottom) Respective responses with nonlinear activation function in the arbitrary color scale in $31 \times 31 \times 328$ pixels (right) to the trained kernels in the first convolution layer in the 3D CNN (left). For visualization purposes, the response maps are resized to match the size of the input images. Here we only show 4 out of 32 such kernels in the first convolution layer.

hydrogen $\bar{x}_{\rm H\,I} = 0.496$ at $z = 7.51$ (the redshift to the very left of the shown light cone) for the benchmark model (middle light cone in both top and bottom panels), but when the parameters are changed to $\log_{10}(T_{\rm vir}) = 5.466$ ($\log_{10}(\zeta) = 1.233$), the reionization is delayed so that $\bar{x}_{\rm H\,I} = 0.938$ ($\bar{x}_{\rm H\,I} = 0.904$) at $z = 7.51$, and when the parameters are changed to $\log_{10}(T_{\rm vir}) = 4.534$ ($\log_{10}(\zeta) = 2.165$), the reionization is accelerated so that $\bar{x}_{\rm H\,I} = 0.015$ ($\bar{x}_{\rm H\,I} = 0.018$) at $z = 7.51$. These comparisons show that the light-cone 21 cm images are more sensitive to parameter $T_{\rm vir}$ than ζ, which explains why the former has better estimation performance with the 3D CNN than the latter.

To provide an interpretation of the features the 3D CNNs have learned, we consider two visualization methods as follows. First, we visualize the trained cubic ($5 \times 5 \times 5$) kernels for the first layer of convolution, as shown in Figure 7. While patterns in the response maps to the kernels are generally difficult to interpret, some patterns might be related to different shapes of H II bubbles. We also find some blank response maps that capture no features of the input images, which might imply that some employed kernels may be unnecessary. We do not show the visualization of kernels in the higher-level convolution layers because it is even harder to interpret the information from there.

Saliency mapping (Simonyan et al. 2014; Springenberg et al. 2014; Zeiler & Fergus 2014; Zhou et al. 2016; Selvaraju et al. 2017) is another useful visualization approach for deep-learning interpretation. In this paper, we calculate the simple gradient-based saliency maps, which are basically the gradients $\partial \boldsymbol{t}/\partial \boldsymbol{d}$ of output data summaries $\boldsymbol{t}$ with respect to the input field $\boldsymbol{d}$. Saliency maps visualize the sensitivity of the outputs to the changes in the inputs. In Figure 8, we plot the saliency maps of the output reionization parameters, $\boldsymbol{t}_i = \log_{10}(T_{\rm vir})$ and $\log_{10}(\zeta)$, respectively, with respect to the input light-cone 21 cm images. For visualization purposes, we plot the logarithmic absolute value of the gradients, which is rescaled to the range [0, 1]. We find that the saliency maps are highlighted in the regions close to the boundaries of H II bubbles, especially at the middle and late stages of reionization.

In summary, these visualizations suggest that the information the 3D CNNs extract from the input light-cone images is likely the shape and boundaries of H II regions where the fields have sharp changes, both in two spatial dimensions and along one frequency dimension of the input map.

3.2. Posterior Inference with the DELFI-3D CNN

In order to test on the estimates from the DELFI-3D CNN with the trained NDEs, we employ a new set of 1000 test samples (different from the test samples for testing the 3D CNN in Section 3.1). From the parameter posterior inferred by DELFI for each of these test samples, we estimate the posterior median[11] of the MCMC chains and statistical uncertainty. We first test the estimation of parameter medians. In Table 2, we show that the coefficient of determination is improved from $R^2 = 0.993$ (0.983) by 3D CNN to 0.997 (0.992) by the DELFI-3D CNN, for $\log(T_{\rm vir})$ ($\log\zeta$). In Figure 9, we show the predicted parameters with 1σ errors versus true parameter value, which shows reasonably small scatters from the perfect (zero-error) recovery. Similar to Figure 5, Figure 10 shows the 2D probability density distribution of the fractional errors ϵ of the parameters estimated by the DELFI-3D CNN with respect to the true values. Within the 68% probability regions in the joint probability density distribution, the fractional error (or the recovery accuracy) is improved from 0.09 (0.12) by 3D CNN to 0.07 (0.06) by the DELFI-3D CNN, for $T_{\rm vir}$ (ζ), as shown in Table 2. These comparisons show that the DELFI-3D CNN can reduce the systematics in the parameter estimation by the 3D CNN.

Now we test the Bayesian inference by the DELFI-3D CNN, a property that cannot be provided by the 3D CNN alone. As a demonstration of concept, we consider two representative mock observations, the "Faint Galaxies Model" and "Bright Galaxies Model", whose definitions are listed as the "True value" in

[11] An alternative point estimate derived from the posterior would be the posterior mean of the MCMC chains. We find that the R^2 value for the posterior median and that for the posterior mean are nearly identical.

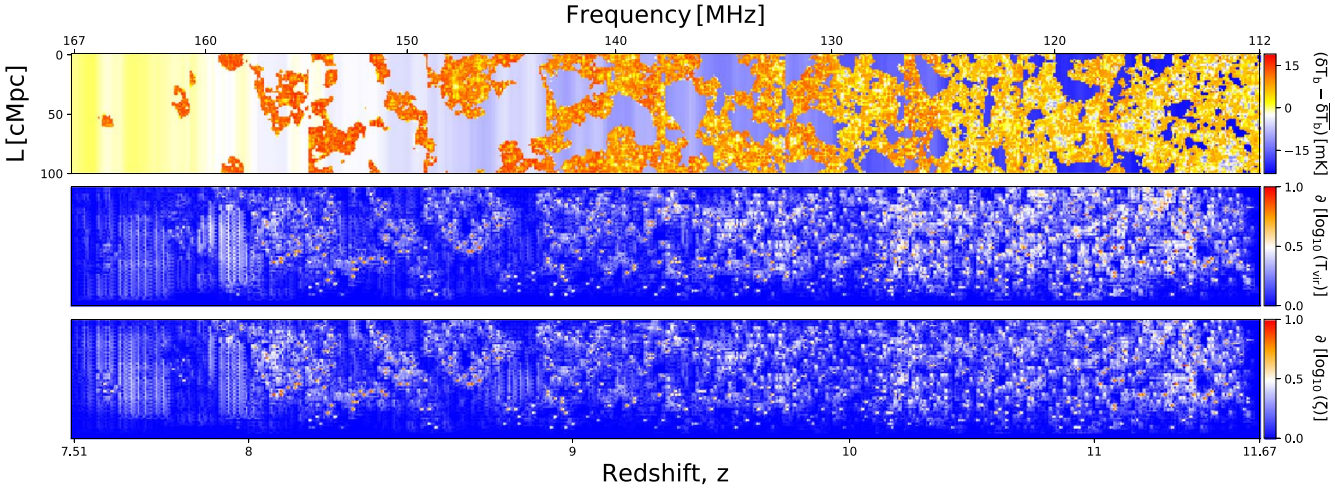

Figure 8. An illustration of saliency maps. (Top) Input light-cone 21 cm images with parameters $\log_{10}(T_{\rm vir}) = 4.101$ and $\log_{10}(\zeta) = 1.389$. (Middle and bottom) Saliency maps of reionization parameters, i.e., the logarithmic absolute value of the gradients of the parameter $\log_{10}(T_{\rm vir})$ and $\log_{10}(\zeta)$, respectively, with respect to the variation in the input image. We rescale the saliency maps to the range [0, 1].

Table 2
Recovery Performance by the 3D CNN and DELFI-3D CNN

		3D CNN	DELFI-3D CNN
R^{2a}	$\log_{10}(T_{\rm vir})$	0.993	0.997
	$\log_{10}(\zeta)$	0.983	0.992
ϵ^{b}	$T_{\rm vir}$	(− 0.09, 0.08)	(− 0.07, 0.07)
	ζ	(− 0.12, 0.08)	(− 0.06, 0.06)

Notes.
[a] The coefficient of determination R^2 is computed for the recovered parameter in logarithmic scale predicted by the networks.
[b] The fractional error ϵ refers to that of the deduced parameter in linear scale within the 68% probability regions in the joint $T_{\rm vir}$-ζ probability density distribution of their accuracies.

Table 3, following Greig & Mesinger (2017). These models are chosen as two examples with extreme parameter values, which however are tuned so that the resulting global reionization histories are both consistent with the latest constraints on the CMB electron scattering optical depth (Planck Collaboration et al. 2016). As a result, their reionization evolutions are similar, as shown in Figure 11. However, the H II bubbles in the Faint Galaxies Model are smaller and more fractal than in the Bright Galaxies Model, because reionization in the former model is powered by more abundant low-mass galaxies yet with smaller ionization efficiency (due to smaller escape fraction of ionizing photons) than in the latter model. In Figure 12, we show the results of posterior inference for both models and list the median and 1σ errors in Table 3. For the former model, the systematic shift (i.e., relative errors of the predicted medians with respect to the true values) and the 1σ statistical errors are $-0.04\% \pm 0.5\%$ ($-0.1\% \pm 1.6\%$), for $\log_{10}(T_{\rm vir})$ ($\log_{10}\zeta$), respectively, or equivalently the systematic shift and 1σ statistical errors are $-0.4\% \pm 5.8\%$ ($-0.3\% \pm 5.4\%$), for $T_{\rm vir}$ (ζ), respectively. For the latter model, the systematic shift and 1σ statistical errors are $0.1\% \pm 0.7\%$ ($0.3\% \pm 1.6\%$) for $\log_{10}(T_{\rm vir})$ ($\log_{10}\zeta$), or equivalently $1.9\%\,^{+9.0\%}_{-8.1\%}$ ($1.4\%\,^{+8.8\%}_{-7.4\%}$) for $T_{\rm vir}$ (ζ), respectively. For both models, the total errors are $\lesssim$10% for $T_{\rm vir}$ and ζ. Note that although for the recoveries in Figure 12 the medians happen to be very close to the true values, in general, the medians can be scattered as large as represented by the statistical errors.

Now that we have the tool to estimate the Bayesian posteriors, we can directly test whether more information from the 3D 21 cm images is indeed exploited than just from the 21 cm power spectrum. For the 21 cm power spectrum Bayesian inference with the MCMC sampling, we employ the publicly available code `21CMMC` (Greig & Mesinger 2015, 2017, 2018).[12] For the setup of `21CMMC`, we generate the mock power spectra at 10 different redshifts, each estimated from a coeval[13] box of 100 comoving Mpc on each side. In order to perform a fair comparison between the results from the DELFI-3D CNN and from the `21CMMC`, we take the implementations as follows. (i) These 10 coeval boxes altogether cover the same redshift range with the same reionization parameters as in the 3D CNN. (ii) The modes in the full k-range corresponding to the box size and cell size are employed in the `21CMMC` analysis. (iii) The same set of random seeds for the initial conditions at 10 different coeval boxes is employed for the tests between the DELFI-3D CNN and `21CMMC`, in order to eliminate any impact of the initial conditions. To construct the likelihood for the analysis based on the power spectrum we model the sample variance from the mock observation by $P_{\rm sv} = P_{21}(k)/\sqrt{N(k)}$, where $P_{21}(k)$ is the 21 cm brightness temperature power spectrum, and $N(k)$ is the number of modes in a k-mode spherical shell. We then perform the Bayesian inference with 200 walkers, using the power spectra at 10 different redshifts. For each walker, we choose the "burn-in chain" number to be 250, and the main chain number to be 3000. The results of Bayesian inference for the Faint Galaxies Model and Bright Galaxies Model are shown in Figure 12 and listed in Table 3. For the former model, the systematic shift and 1σ statistical error are $-6\%\,^{+7.9\%}_{-5.5\%}$ ($-7\%\,^{+6.1\%}_{-5.4\%}$), for $T_{\rm vir}$ (ζ), respectively. For the latter model, the

[12] https://github.com/BradGreig/21CMMC
[13] Strictly speaking, power spectra from the *light-cone* boxes should be employed. However, the light-cone effect is only nonnegligible at large scales (Datta et al. 2012, 2014), so power spectrum analysis from the coeval boxes, which greatly reduces the computational costs, does not significantly change the inference results. Since the 21 cm power spectrum analysis is not the focus of our paper but just serves for comparison, we choose to use coeval boxes for `21CMMC` herein.

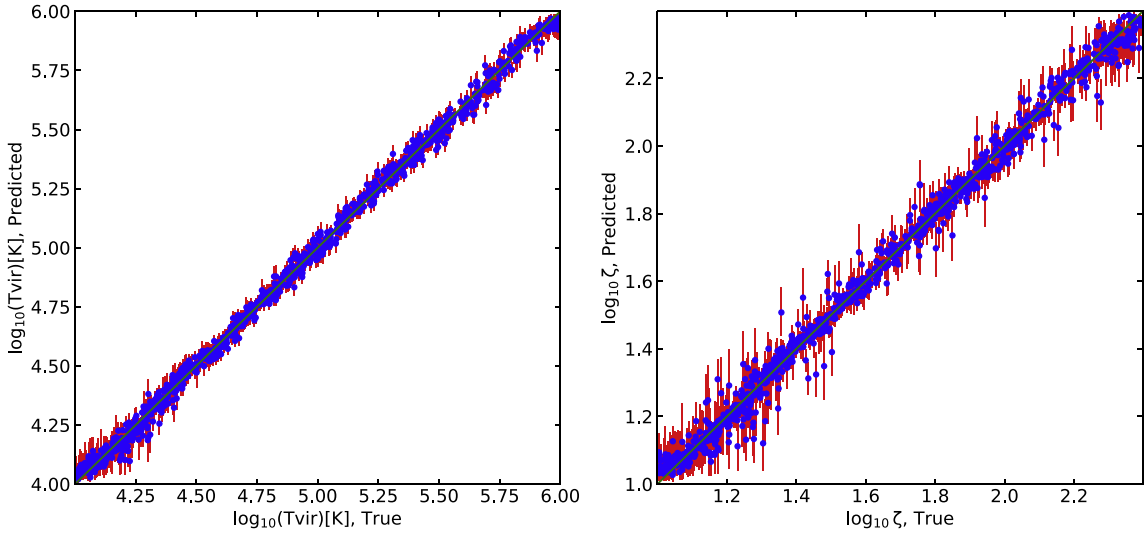

Figure 9. Reionization parameter recovery performance with the DELFI-3D CNN using 1000 new test samples. Shown is the predicted value $y_{\rm pred}$ vs. the true value $y_{\rm true}$ (blue dots) of each parameter $y = \log(T_{\rm vir})$ (left) and $\log(\zeta)$ (right), respectively, with 1σ error inferred by the DELFI-3D CNN (red error bars). The green diagonal line indicates the perfect (zero-error) recovery.

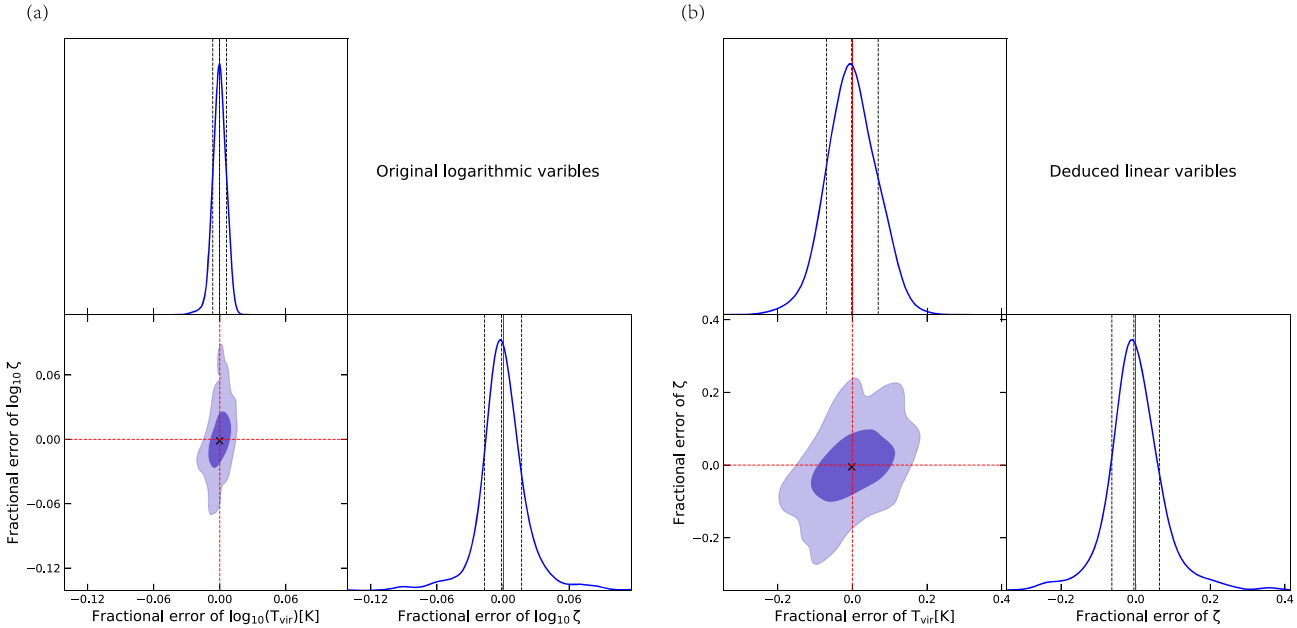

Figure 10. Same as Figure 5 but for testing the DELFI-3D CNN with 1000 new test samples.

Table 3
Bayesian Inference with the DELFI-3D CNN and with the 21CMMC

	Faint Galaxies Model			Bright Galaxies Model		
Parameter	True value	DELFI-3D CNN	21CMMC	True value	DELFI-3D CNN	21CMMC
$\log_{10}(T_{\rm vir}/{\rm K})$	4.699	$4.697^{+0.024}_{-0.024}$	$4.673^{+0.033}_{-0.025}$	5.477	$5.485^{+0.037}_{-0.036}$	$5.435^{+0.050}_{-0.052}$
$\log_{10}(\zeta)$	1.477	$1.475^{+0.023}_{-0.023}$	$1.444^{+0.026}_{-0.023}$	2.301	$2.307^{+0.036}_{-0.033}$	$2.226^{+0.077}_{-0.072}$
$T_{\rm vir}(\times 10^5{\rm K})$	0.5	$0.498^{+0.029}_{-0.027}$	$0.471^{+0.037}_{-0.026}$	3	$3.056^{+0.269}_{-0.243}$	$2.724^{+0.331}_{-0.310}$
ζ	30	$29.9^{+1.6}_{-1.5}$	$27.8^{+1.7}_{-1.5}$	200	$202.8^{+17.6}_{-14.8}$	$168.2^{+32.5}_{-25.6}$

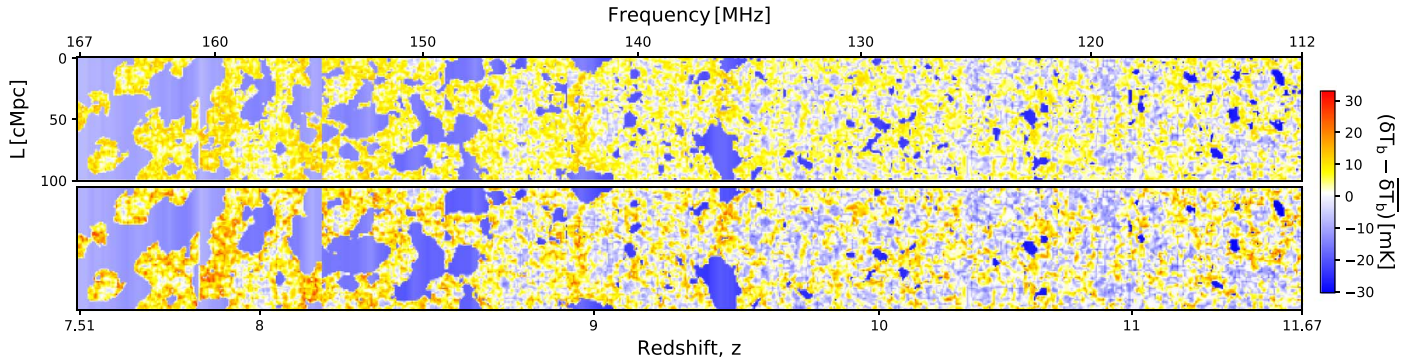

Figure 11. An illustration of the light-cone 21 cm images in two reionization models—the "Faint Galaxies Model" (top) and the "Bright Galaxies Model" (bottom), with their true parameter values listed in Table 3.

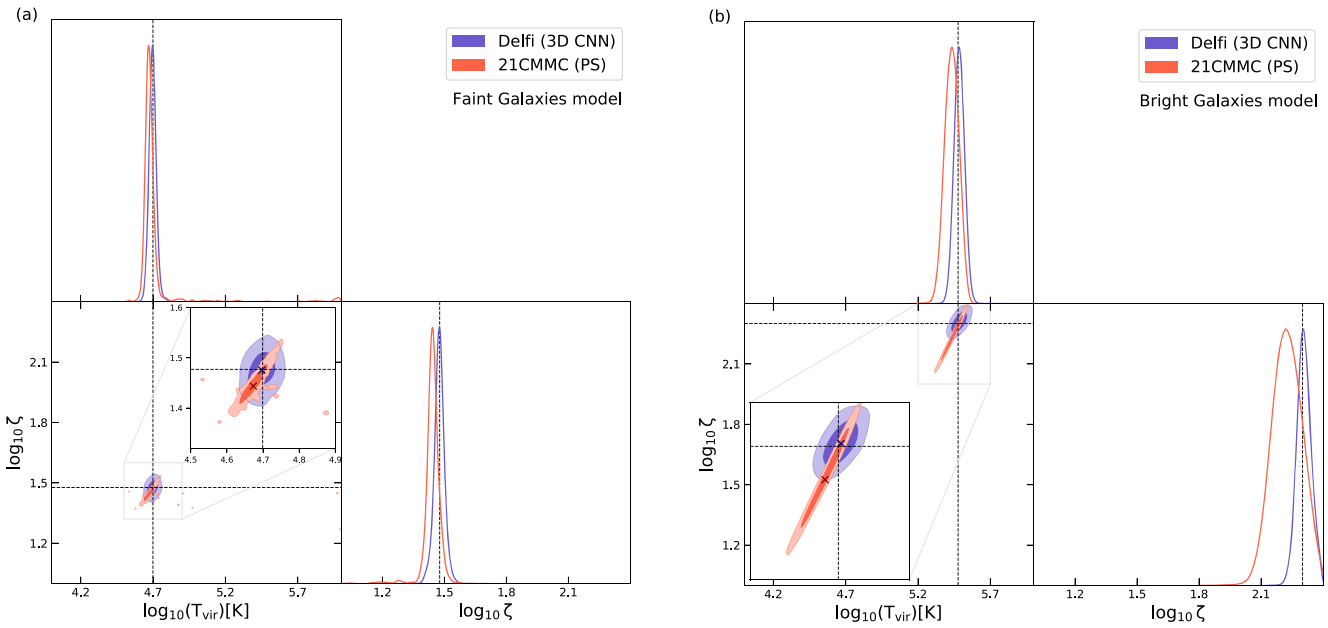

Figure 12. The posteriors estimated by the DELFI-3D CNN (blue) for two mock observations, the "Faint Galaxies Model" (left) and the "Bright Galaxies Model" (right). We show the median (cross), the 1σ (dark blue), and 2σ (light blue) confidence regions. For comparison, we also show the posteriors estimated by the 21 cm power spectrum analysis with MCMC sampling (red). The dashed lines indicate the true parameter values.

systematic shift and 1σ statistical error are $-9\%^{+12.2\%}_{-11.4\%}$ ($-16\%^{+19.3\%}_{-15.2\%}$) for $T_{\rm vir}$ (ζ), respectively. We have also checked the convergence of the MCMC chains as shown in Appendix F. In comparison, the DELFI-3D CNN outperforms the `21CMMC` both in the median estimation and in the statistical errors of Bayesian inference. This implies that the DELFI-3D CNN may take advantage of more, if not all, information in the 3D 21 cm images than just in the power spectrum. We should also note that the computational speed of DELFI for each inference is 2 orders of magnitude faster than `21CMMC`, which makes this approach practically more useful in the futuristic 21 cm data analysis.

We further evaluate the accuracy of the posteriors over 1000 test samples and follow the recent work of Ho et al. (2021) to perform the statistical check. Similar evaluation methods could be found in Kodi Ramanah et al. (2021), Perreault Levasseur et al. (2017), and Wagner-Carena et al. (2021). In Figure 13, we show the posterior calibration plot, which we briefly describe below. For a given percentage p of the probability volume, the fraction of samples with the true parameter values falling into this probability volume of the inferred posterior is defined as *empirical percentage* (*ep*),

$$ep = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[X_i < \hat{X}_{p,i}],$$

where $\mathbf{1}[\boldsymbol{A}]$ is the indicator function, which returns 1 if the condition $\boldsymbol{A}$ is satisfied and 0 otherwise. N is the total number of evaluation samples. Here X_i is the true value of the ith sample. $\hat{X}_{p,i}$ is the percentile corresponding to percentage p of the inferred posterior $\hat{x}_i$ of the ith sample, i.e., $P(\hat{X} < \hat{X}_p | \hat{X} \in \hat{x}) = p$. If the inferred uncertainties are perfectly accurate, then $ep = p$. For univariate posteriors, if the condition $ep > p$ ($ep < p$) always holds, then the inference biases toward a larger (smaller) value. If the condition $ep > p$ ($ep < p$) holds when $p < 0.5$ and the condition $ep < p$ ($ep > p$) holds when $p > 0.5$, then the inference underestimates (overestimates) the uncertainties.

Figure 13 shows that the uncertainty estimations for both parameters are highly accurate, with only small deviations from the perfect calibration. For both parameters, the inferred posteriors are a slight overestimation of uncertainties. In other words, our uncertainty estimation is on the conservative side.

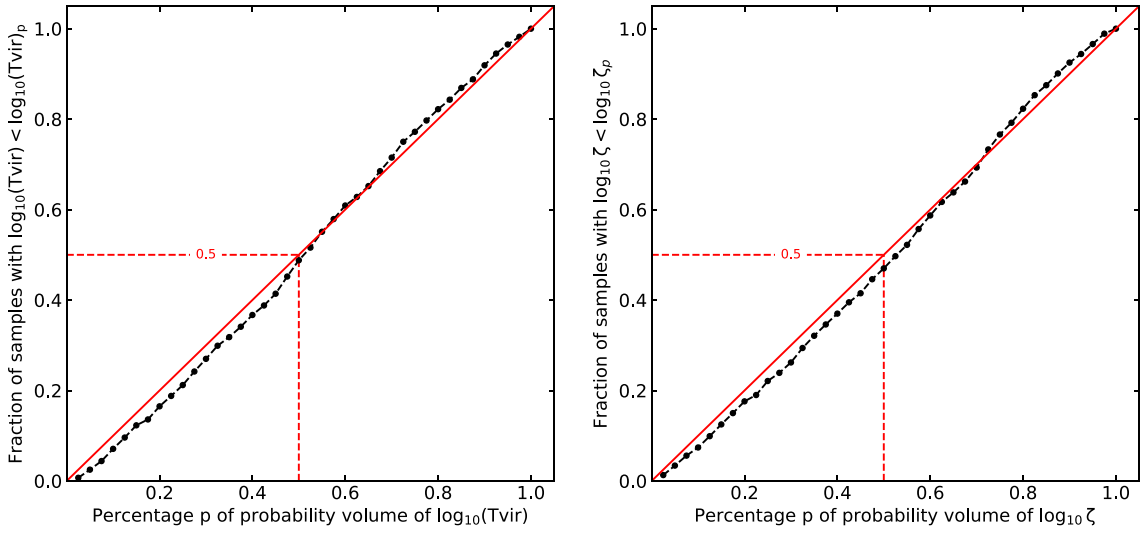

Figure 13. Posterior calibration plot, i.e., the empirical percentage *ep* vs. the percentage *p* of the probability volume for the parameters $\log_{10}(T_{\rm vir})$ (left) and $\log_{10}(\zeta)$ (right). Shown are the actual calibration with 1000 new test samples (black dots) and the perfect calibration (red solid line).

This trend, as also found in Kodi Ramanah et al. (2021; see their Figure 6) and in Ho et al. (2021; see their Section 4), may result from the nonoptimal data compression of the trained 3D CNN.

4. Summary

In this paper, we perform a Bayesian inference of the reionization parameters where the likelihood is implicitly defined through forward simulations using DELFI (specifically the `pydelfi` implementation). The information in the 3D 21 cm images is exploited by compressing the 3D image data into informative summaries with a trained 3D CNN. While these summaries are interpretable as point estimates of the reionization parameters, they are technically an intermediate step used as input to the full posterior inference of DELFI. We show that this method (DELFI-3D CNN) recovers accurate posterior distributions for the reionization parameters, estimating the posterior median $T_{\rm vir}$ (ζ) with a relative error of 7% (6%) with respect to the true parameter value, in most (68%) of the test samples.

This level of recovery accuracy improves upon the earlier analysis based on 2D CNN (G19). In their work, the relative error of the parameter $\log_{10}(T_{\rm vir})$ is $\lesssim$1%, which is comparable to ours, but the relative error of the parameter ζ is $\sim$10%. While the total light-cone volume therein is much larger than ours (300 versus 100 cMpc per each transverse side to the LOS), the 2D study only extracted five slices along the LOS for each light cone. In other words, for the same LOS redshift interval, the data volume exploited in our 3D CNN study is $100^2/(5\times 300)\approx 6.7$ times larger than in the previous 2D study. This explains qualitatively why the DELFI-3D CNN recovery outperforms the 2D CNN study.

For the purpose of comparison, we perform an MCMC analysis of the 21 cm power spectrum alone based on the 3D light cone and using a Gaussian likelihood approximation. This power spectrum analysis with the `21CMMC` results in less accurate credible parameter regions than inferred by the DELFI-3D CNN, both in terms of the location and shape of the contours.

These show that the DELFI-3D CNN exploits more information in the 3D 21 cm images than a 2D CNN or just the power spectrum analysis by `21CMMC`.

The posterior calibration shows that the uncertainties inferred by the DELFI-3D CNN are statistically self-consistent, but we caution that it is likely that the uncertainties are slightly overestimated.

As a demonstration of concept, this paper only considers the ideal case in which the thermal noise and residual foregrounds are neglected. The DELFI framework is flexible for incorporating these realistic effects, as will be done in a follow-up paper, so this method will be a promising approach for the scientific interpretation of future 21 cm observation data. We also note that, in principle, the 3D CNN can be replaced by other compressors that optimize a trade-off between image compression and information extraction. This provides room for improving the performance of posterior inference, which will be considered in a follow-up paper.

This work is supported by National SKA Program of China (grant No. 2020SKA0110401), NSFC (grant No. 11821303), and National Key R&D Program of China (grant No. 2018YFA0404502). BDW acknowledges support from the Simons Foundation. We thank Yangyao Chen, Rui Huang, Benoit Semelin, Hayato Shimabukuro, and Meng Zhou for useful discussions and helps. The deep learning computations and reionization simulations were ran in the R2D2 and Metis GPU workstations and at the Venus cluster at the Tsinghua University, respectively.

Software: 21CMMC (Greig & Mesinger 2015, 2017, 2018), 21cmFAST (Mesinger & Furlanetto 2007; Mesinger et al. 2011), pydelfi (Alsing et al. 2019), Keras (Chollet et al. 2015), GetDist (Lewis 2019), NumPy (Harris et al. 2020), Matplotlib (Hunter 2007), SciPy (Virtanen et al. 2020), scikit-learn (Pedregosa et al. 2011), Python2 (Van Rossum & Drake 1995), Python3 (Van Rossum & Drake 2009).

Appendix A
Optimization of 3D CNN

Improving the performance of deep learning requires significant tuning. While there is no standard process to achieve the absolutely "best" performance, we now reproduce some elements of good practice for tuning to achieve an optimized model that have emerged. Interested readers are referred to Goodfellow et al. (2016) for the techniques of network optimization. First, we can improve the performance by preprocessing the data. For example, in order to avoid large values in the networks, we can standardize the data to the range (0,1) (normalization) or (−1,1) (rescaling). Second, once the data are in place, the hyperparameters are tuned, e.g., initializing network weights, changing the learning rates, and experimenting with different minibatch sizes and training epochs.

In this section, we explore the optimization of the 3D CNN by considering the variations in the network weights, parameter ranges, and sampling strategies and investigate their impacts on the performance of parameter estimation, as listed in Table 4. The fiducial setup is the one adopted in the main part of this paper. Case A setup focuses on the change in the relative weight of two output parameters, i.e., increasing (decreasing) the network weight of $\log_{10}(T_{\rm vir})$ from 0.8 in the fiducial setup to 1.0 (0.6) in Case A1 (A2) when fixing the weight of $\log_{10}(\zeta)$ to be 1.0. The network weight ratio of 1:1 means equal importance for two output parameters during training, while the network weight ratio of 0.6:1 means that fitting to $\log_{10}(T_{\rm vir})$ is assigned less weight than fitting to $\log_{10}(\zeta)$. Since the 21 cm signal depends on the output parameters with different sensitivities, a proper adjustment of their network weights can optimize the overall performance of parameter estimation. We find that the weight ratio of 0.8:1 (fiducial setup) gives the best performance results in comparison to Case A1 and A2.

Next, we consider the effect of the parameter range set a priori. The range of ζ is shrunk from $10 \leqslant \zeta \leqslant 250$ in the fiducial setup to $10 \leqslant \zeta \leqslant 100$ in Case B, while fixing the range of $T_{\rm vir}$ as $10^4 \leqslant T_{\rm vir} \leqslant 10^6$ K. Table 4 shows that the accuracies of parameter estimation in the fiducial and Case B setups are very similar.

Third, we consider the sampling of ζ in the logarithmic scale (Case B) and in the linear scale (Case C) with the same range of ζ, while fixing the sampling of $T_{\rm vir}$ in the logarithmic scale. Case C has the same sampling strategy as in G19. Table 4 shows that the accuracies of parameter estimation in the Case B and Case C setups are very similar. Note that in Case B, $R^2 = 0.970$ for $\log_{10}\zeta$ corresponds to $R^2 = 0.955$ for ζ, which is almost the same result as in Case C.

We conclude that different parameter ranges or sampling strategies do not significantly affect the overall performance of parameter estimation with the 3D CNN. However, the performance can be optimized by fine-tuning the relative output weights between different parameters. The fiducial setup in this paper results from this optimization process.

Appendix B
Effect of Missing the $k_\perp = 0$ Mode in the Interferometer Measurements

In our preparation of mock light-cone images, we subtract the mean of the 2D slice from the 21 cm signal, because the interferometer cannot measure the $\boldsymbol{k}_\perp = 0$ mode. However, in G19, this step was skipped. To remove possible systematics due to this effect when comparing the results of G19 and ours, we perform a test in which the light-cone images are prepared without the subtraction step (see the lower panel of Figure 2), as in G19. We test the reionization parameter recovery performance by comparing the predicted parameter values from the trained 3D CNN with the true values in Figure 14 and plotting the 2D joint probability density distribution of the fractional errors for the parameters $T_{\rm vir}$ and ζ in Figure 15.

The R^2 score is 0.992 (0.972) for $\log(T_{\rm vir})$ ($\log\zeta$). Within the 68% probability regions in the joint probability density distribution, the fractional error (or the recovery accuracy) is (−0.12, 0.07) for $T_{\rm vir}$ and (−0.13, 0.13) for ζ, respectively. In comparison with the 3D CNN results shown in Table 2, the performance of both parameters in this case is similar to (but

Table 4
Variations of Network Setups in the 3D CNN and their Results of Reionization Parameter Estimation

Model			Fiducial	Case A1	Case A2	Case B	Case C
Setup		Weight for $\log_{10}(T_{\rm vir})$[a]	0.8	1.0	0.6	0.8	0.8
		Parameter Range for ζ[b]	$1 \leqslant \log_{10}\zeta \leqslant 2.398$	$1 \leqslant \log_{10}\zeta \leqslant 2.398$	$1 \leqslant \log_{10}\zeta \leqslant 2.398$	$1 \leqslant \log_{10}\zeta \leqslant 2$	$10 \leqslant \zeta \leqslant 100$
		Output Type for ζ[c]	logarithmic	logarithmic	logarithmic	logarithmic	linear
Results	R^2[d]	$\log_{10}(T_{\rm vir})$	0.993	0.991	0.989	0.995	0.991
		$\log_{10}(\zeta)$	0.983	0.970	0.977	0.970	0.951
	ϵ[e]	$T_{\rm vir}$	(−0.09, 0.08)	(−0.12, 0.11)	(−0.14, 0.10)	(−0.08, 0.09)	(−0.15, 0.08)
		ζ	(−0.12, 0.08)	(−0.12, 0.14)	(−0.11, 0.11)	(−0.09, 0.08)	(−0.10, 0.08)

Notes.
[a] The network weight for $\log_{10}\zeta$ (or ζ in Case C) is fixed as 1.0.
[b] The parameter range for $T_{\rm vir}$ is fixed as $4 \leqslant \log_{10}(T_{\rm vir}/{\rm K}) \leqslant 6$.
[c] "Logarithmic" ("linear") for ζ refers to the output of $\log_{10}\zeta$ (ζ), respectively. The output for $T_{\rm vir}$ is always logarithmic, i.e., $\log_{10}(T_{\rm vir})$.
[d] The coefficient of determination R^2 is computed for the recovered parameter in the logarithmic scale predicted from the networks, except for Case C where $R^2(\zeta)$ is for ζ in the linear scale.
[e] The fractional error ϵ refers to that of the deduced parameter in the linear scale within the 68% probability regions in the joint $T_{\rm vir}$–ζ probability density distribution of their accuracies.

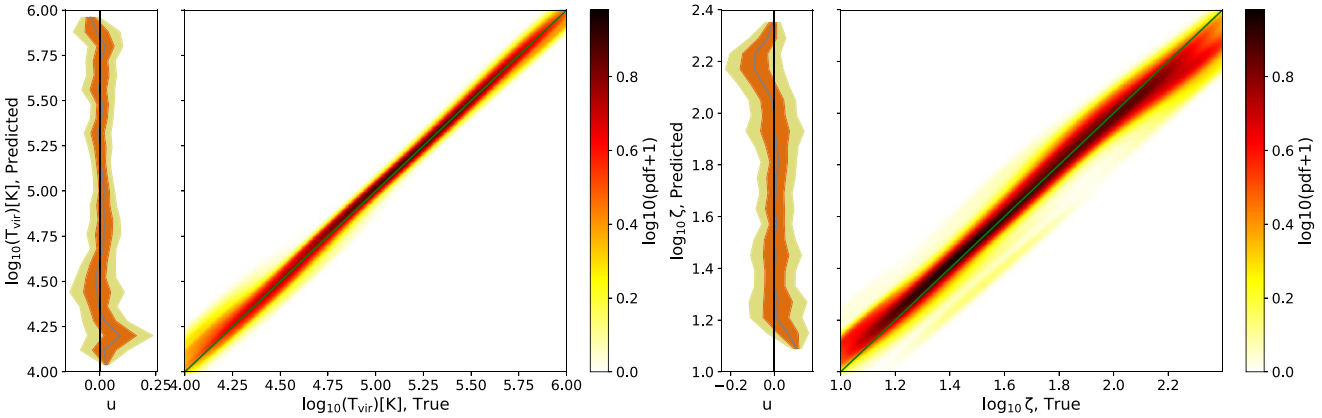

Figure 14. Same as Figure 4 but for the case in which the light-cone images are the 21 cm brightness temperature without subtraction of the mean of the 2D slice.

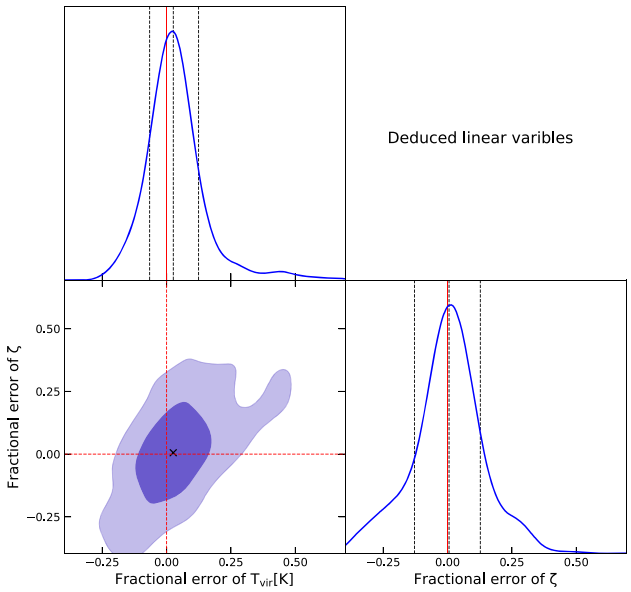

Figure 15. Same as the right panel of Figure 5 but for the case in which the light-cone images are the 21 cm brightness temperature without subtraction of the mean of the 2D slice.

slightly worse[14] than) that in the fiducial setup of the 3D CNN. In other words, not removing the $\boldsymbol{k}_\perp = 0$ mode does not affect the performance significantly.

Appendix C
Formalism of NDEs

C.1. MDNs

For the typical Gaussian MDN networks, the density estimators can be defined as:

$$p(\boldsymbol{t}|\boldsymbol{\theta};\boldsymbol{w}) = \sum_{k=1}^{n_c} r_k(\boldsymbol{\theta};\boldsymbol{w}) \times \mathcal{N}[\boldsymbol{t}|\boldsymbol{\mu}_k(\boldsymbol{\theta};\boldsymbol{w}), \boldsymbol{C}_k \equiv \boldsymbol{\Sigma}_k(\boldsymbol{\theta};\boldsymbol{w})\boldsymbol{\Sigma}_k^T(\boldsymbol{\theta};\boldsymbol{w})], \tag{C1}$$

where n_c is the number of Gaussian components. Each component has three properties—weights of different components $r_k(\boldsymbol{\theta};\boldsymbol{w})$, means $\boldsymbol{\mu}_k(\boldsymbol{\theta};\boldsymbol{w})$, and covariance factors $\boldsymbol{\Sigma}_k(\boldsymbol{\theta};\boldsymbol{w})$. All of these properties are functions of $\boldsymbol{\theta}$ and the network weights $\boldsymbol{w}$ and different properties require different nonlinear activation functions in the output layers of the networks (A19). Here the function $\mathcal{N}$ is a multivariate Gaussian distribution.

C.2. MAFs

For the vector $\boldsymbol{t}$ following a conditional density $p(\boldsymbol{t}|\boldsymbol{\theta})$, we can express $\boldsymbol{t}$ as a transformation T of $\boldsymbol{z}$ sampled from a base density π:

$$\boldsymbol{t} = T(\boldsymbol{z}) \quad \text{where} \quad \boldsymbol{z} \sim \pi(\boldsymbol{z}|\boldsymbol{\theta}). \tag{C2}$$

If T is invertible and both T and T^{-1} are differentiable, the conditional density of $p(\boldsymbol{t}|\boldsymbol{\theta})$ can be calculated as

$$p(\boldsymbol{t}|\boldsymbol{\theta}) = \pi(\boldsymbol{z}|\boldsymbol{\theta}) \left| \det\left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{t}}\right) \right| = \pi(T^{-1}(\boldsymbol{t},\theta)|\boldsymbol{\theta}) \left| \det\left(\frac{\partial T^{-1}(\boldsymbol{t},\boldsymbol{\theta})}{\partial \boldsymbol{t}}\right) \right|. \tag{C3}$$

The MAFs, as depicted in Figure 16, are blocks of single transformations. We first look at a single block of transformation and simplify the input and output vectors as $\boldsymbol{t}$ and $\boldsymbol{z}$, respectively. The MADE with the network weights $\boldsymbol{w}$ expresses the conditional density in an autoregressive way $p(\boldsymbol{t}|\boldsymbol{\theta};\boldsymbol{w}) = \prod_{i=1}^{\dim(\boldsymbol{t})} p(t_i|t_{1:i-1},\boldsymbol{\theta};\boldsymbol{w})$ and outputs the mean μ_i and variance σ_i of the 1D Gaussian distribution $p(t_i|t_{1:i-1},\boldsymbol{\theta};\boldsymbol{w})$:

$$\mu_i = f_{\mu_i}(t_{1:i-1},\boldsymbol{\theta};\boldsymbol{w}),\ \sigma_i = f_{\sigma_i}(t_{1:i-1},\boldsymbol{\theta};\boldsymbol{w}). \tag{C4}$$

Here f represents the MADE.

An affine transformation can be applied to transform $\boldsymbol{t}$ to another vector $\boldsymbol{z}$ with each component

$$z_i = (t_i - \mu_i)/\sigma_i. \tag{C5}$$

[14] Note that the optimization of networks upon the new data set in the case of not removing the $\boldsymbol{k}_\perp = 0$ mode may be different from the fiducial setup. Since this case is not real, we do not tune the networks with greater efforts.

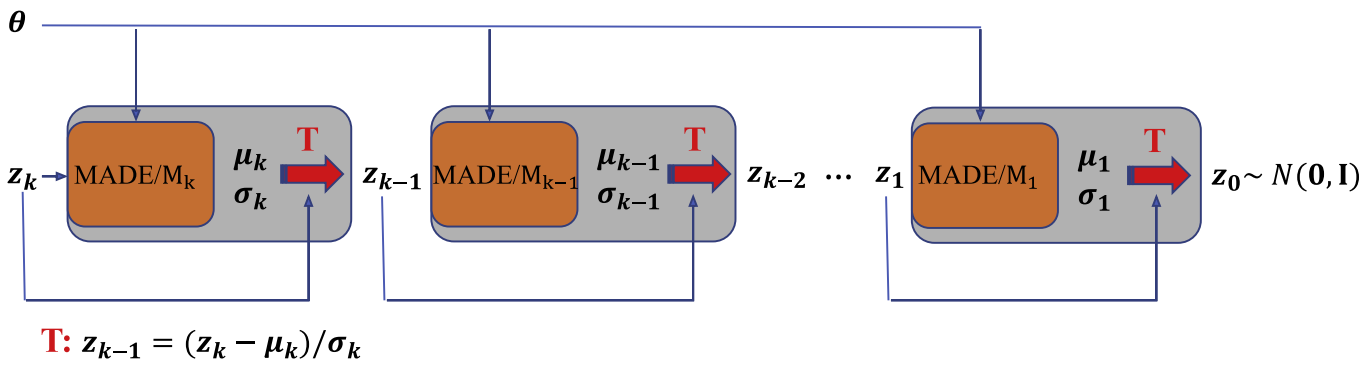

Figure 16. An illustration of the masked autoregressive flows (MAFs). It transforms the data summary $\boldsymbol{t}$ back to a vector z_0 following a normal distribution. For a single transformation, the MADE takes a vector z_k and parameter $\boldsymbol{\theta}$ as the input and outputs the mean μ_k and variance σ_k of the autoregressive 1D distributions. Then an affine transformation transforms z_k into another vector z_{k-1}. Here z_k refers to the input data summary $\boldsymbol{t}$ used in this paper.

In this way, the Jacobin of the transformation is triangular, and the absolute value of determinant can be calculated by

$$\left| \det\left(\frac{\partial T^{-1}(\boldsymbol{t}, \boldsymbol{\theta}; \boldsymbol{w})}{\partial \boldsymbol{t}} \right) \right| = \prod_{i=1}^{\dim(\boldsymbol{t})} (1/\sigma_i(\boldsymbol{t}, \boldsymbol{\theta}; \boldsymbol{w})). \quad \text{(C6)}$$

The composition of single transformations, $T = T_k \circ T_{k-1} \circ \ldots \circ T_1$, is also invertible, and T and T^{-1} are differentiable. By stacking multiple blocks of transformations, the output of the previous transformation z_{k-1} is used as the input of the next transformation. The last transformation outputs the vector z_0 that follows the base density, which is taken to be a multivariate Gaussian distribution. The final conditional density can be written as

$$p(\boldsymbol{t}|\boldsymbol{\theta}; \boldsymbol{w}) = \mathcal{N}[z_0(\boldsymbol{t}, \boldsymbol{\theta}; \boldsymbol{w})|\mathbf{0}, \boldsymbol{I}] \times \prod_{n=1}^{N_{\rm mades}} \prod_{i=1}^{\dim(\boldsymbol{t})} (1/\sigma_i^n(\boldsymbol{t}, \boldsymbol{\theta}; \boldsymbol{w})), \quad \text{(C7)}$$

where the $N_{\rm mades}$ and $\dim(\boldsymbol{t})$ are the number of MADEs and the dimension of vector $\boldsymbol{t}$, respectively.

C.3. Training of NDEs

Training NDEs is to fit the estimators $p(\boldsymbol{t}|\boldsymbol{\theta}; \boldsymbol{w})$ to a target distribution $p^{\star}(\boldsymbol{t}|\boldsymbol{\theta})$ by minimizing the Kullback–Leibler divergence between them,

$$D_{\rm KL}(p^*|p) = \int p^*(\boldsymbol{t}|\boldsymbol{\theta}) \ln\left(\frac{p(\boldsymbol{t}|\boldsymbol{\theta}; \boldsymbol{w})}{p^*(\boldsymbol{t}|\boldsymbol{\theta})} \right) \mathrm{d}\boldsymbol{t}. \quad \text{(C8)}$$

In practice, we can use a Monte Carlo estimate (A19).

Appendix D
Effect of Initial Conditions

The 21 cm brightness temperature fields evolve from the initial Gaussian random density fields and therefore are affected

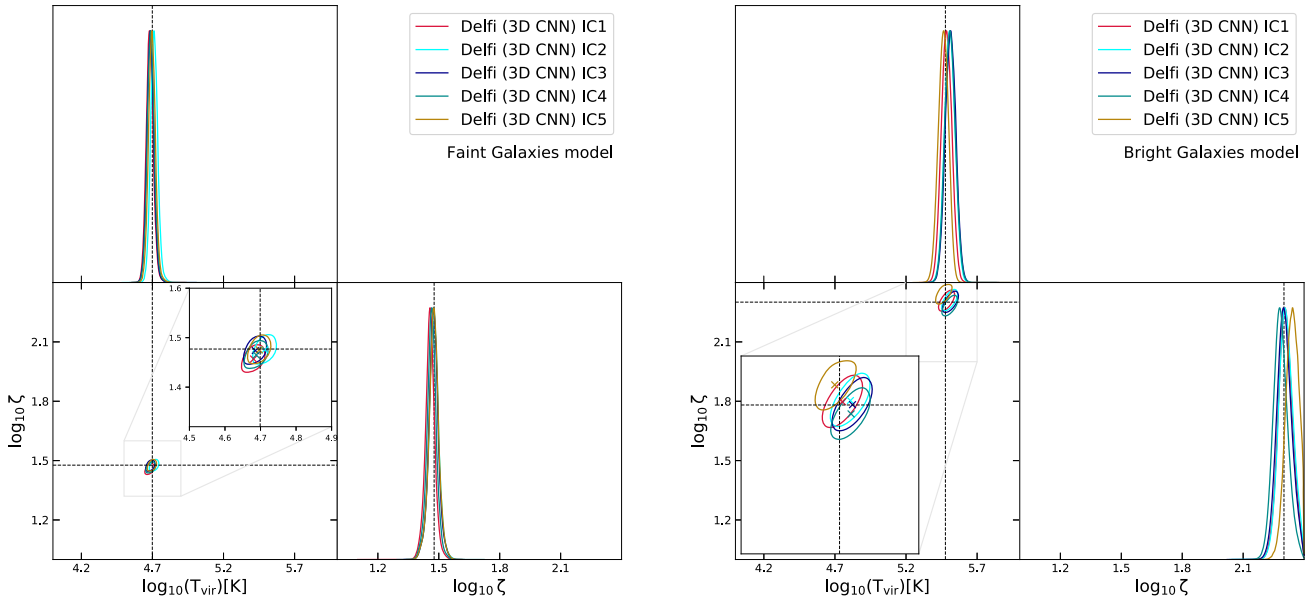

Figure 17. The effect of initial conditions on the posterior inference. Shown are five posteriors (in different colors) estimated by the DELFI-3D CNN from the light-cone images with different initial conditions (ICs), for two mock observations, the "Faint Galaxies Model" (left) and the "Bright Galaxies Model" (right). We show the median (cross) and 1σ confidence region (contour). The dashed lines indicate the true parameter values.

by cosmic variance. In this section, we investigate the impact of different initial conditions of realizations on the parameter recovery with the DELFI-3D CNN. In Figure 17, we plot the posterior inferences from five different light-cone images, corresponding to different initial conditions but with the same reionization parameters, for two representative models. We find that, for the Faint Galaxies Model, the 1σ confidence regions from different initial conditions agree with each other almost completely. For the Bright Galaxies Model, the 1σ confidence regions show some scatters, but the true values are always within the 1σ region. We conclude that the posterior inference is mostly robust against the variations in initial conditions due to cosmic variance.

Appendix E
Stabilization of different NDEs

The default setup of NDE in this paper is a stacked ensemble of individual NDEs, including four MDNs and one MAF. The resulting posterior, therefore, is stacked from individual posteriors with weights according to the training loss of the corresponding NDEs. In this section, we test on the robustness of posterior inference with different ensembles of NDEs. In Figure 18, we show the stacked posterior and the individual posteriors for the bright galaxy model. We find that the 1σ confidence region inferred with different NDEs agree with each other very well. We conclude that the posterior inference is robust against the variations in the NDEs.

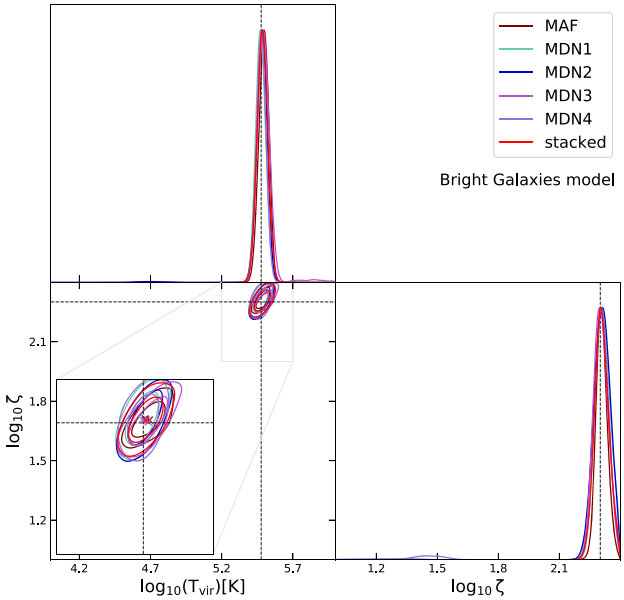

Figure 18. Stabilization of different NDEs. Shown are the posteriors by the DELFI-3D CNN with the individual NDEs (four MDNs and one MAF, with different colors) and with the stacked NDE (red), for the bright galaxies model. We show the median (cross) and 1σ confidence region (contour). The dashed lines indicate the true parameter values.

Appendix F
Convergence of `21CMMC`

In Figure 19, we make a convergence test for the `21CMMC` run, by showing the trajectories of 200 walkers for each mock

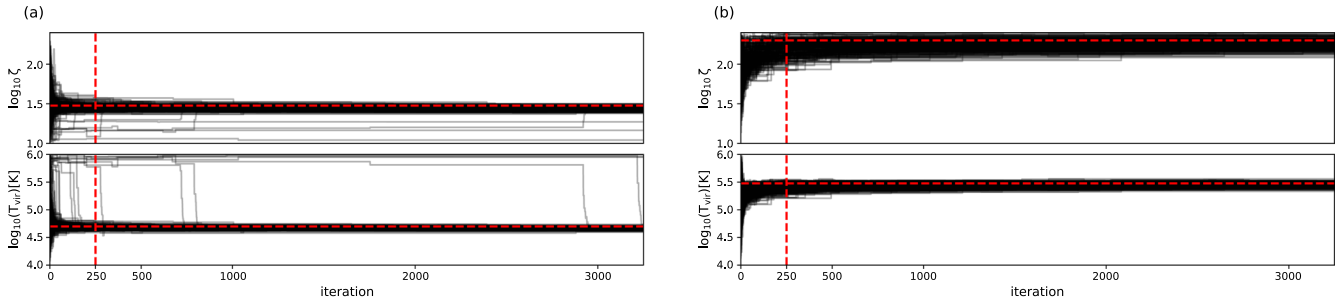

Figure 19. MCMC convergence test for `21CMMC` for two mock observations—the "Faint Galaxies Model" (left) and the "Bright Galaxies Model" (right). For a given mock observation, we use 200 walkers, each with independent trajectory (black lines). We discard the first 250 iterations—the so-called "burn-in" of MCMC chains as indicated by the vertical red dashed lines—and only use the following 3000 iterations to generate the `21CMMC` results. The true parameter values are shown by the horizontal red dashed lines.

observation. The trajectories represent the updating process of the parameters used to explore the likelihood at the two mock observations. For each walker, we discard the initial 250 iterations ("burn-in") and adopt the subsequent 3000 iterations to generate the `21CMMC` results. For both models, the trajectories reach the convergence within the burn-in iterations, which means that the `21CMMC` results are converged.

ORCID iDs

Xiaosheng Zhao https://orcid.org/0000-0002-8328-1447
Yi Mao https://orcid.org/0000-0002-1301-3893
Benjamin D. Wandelt https://orcid.org/0000-0002-5854-8269

References

Akeret, J., Refregier, A., Amara, A., Seehars, S., & Hasner, C. 2015, JCAP, 2015, 043
Alsing, J., Charnock, T., Feeney, S., & Wand elt, B. 2019, MNRAS, 488, 4440
Alsing, J., Wandelt, B., & Feeney, S. 2018, MNRAS, 477, 2874
Amidi, A., Amidi, S., Vlachakis, D., et al. 2018, PeerJ, 6, e4750
Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., & Baskurt, A. 2011, in Human Behavior Understanding, ed. A. A. Salah & B. Lepri (Berlin Heidelberg: Springer), 29
Baldi, P., & Sadowski, P. J. 2013, in Proc. of the 26th Int. Conf. on Neural Information Processing Systems 26 (Red Hook, NY: Curran Associates), 2814
Beane, A., & Lidz, A. 2018, ApJ, 867, 26
Becker, G. D., Bolton, J. S., Madau, P., et al. 2015, MNRAS, 447, 3402
Bishop, C. 1994, Mixture Density Networks, Tech. Rep. NCRG/94/004, Aston University, https://publications.aston.ac.uk/id/eprint/373/
Bonassi, F. V., You, L., & West, M. 2011, Stat. Appl. Genet. Mol., 10, 1
Bouwens, R. J., Illingworth, G. D., Oesch, P. A., et al. 2015, ApJ, 811, 140
Cameron, E., & Pettitt, A. N. 2012, MNRAS, 425, 44
Carassou, S., de Lapparent, V., Bertin, E., & Le Borgne, D. 2017, A&A, 605, A9
Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. 2016, in Medical Image Computing and Computer-Assisted Intervention—MICCAI 2016, 424 ed. S. Ourselin et al. (Cham: Springer Int. Publishing)
Chen, Z., Xu, Y., Wang, Y., & Chen, X. 2019, ApJ, 885, 23
Chollet, F. 2015, Keras, https://github.com/fchollet/keras, GitHub
Dahl, G. E., Sainath, T. N., & Hinton, G. E. 2013, in 2013 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (Vancouver: IEEE), 8609
Datta, K. K., Jensen, H., Majumdar, S., et al. 2014, MNRAS, 442, 1491
Datta, K. K., Mellema, G., Mao, Y., et al. 2012, MNRAS, 424, 1877
Davies, F. B., Hennawi, J. F., Eilers, A.-C., & Lukić, Z. 2018, ApJ, 855, 106
Dayal, P., & Ferrara, A. 2018, PhR, 780, 1
DeBoer, D. R., Parsons, A. R., Aguirre, J. E., et al. 2017, PASP, 129, 045001
Doussot, A., Eames, E., & Semelin, B. 2019, MNRAS, 490, 371
Fan, X., Strauss, M. A., Becker, R. H., et al. 2006, AJ, 132, 117
Fan, Y., Nott, D. J., & Sisson, S. A. 2013, Stat, 2, 34
Finkelstein, S. L., D'Aloisio, A., Paardekooper, J.-P., et al. 2019, ApJ, 879, 36
Furlanetto, S. R., Hernquist, L., & Zaldarriaga, M. 2004a, MNRAS, 354, 695
Furlanetto, S. R., Oh, S. P., & Briggs, F. H. 2006, PhR, 433, 181
Furlanetto, S. R., Zaldarriaga, M., & Hernquist, L. 2004b, ApJ, 613, 1
Gal, Y., & Ghahramani, Z. 2016, in Proc. of Machine Learning Research, 48, Proc. of The 33rd Int. Conf. on Machine Learning, ed. M. F. Balcan & K. Q. Weinberger (New York: PMLR), 1050
Germain, M., Gregor, K., Murray, I., & Larochelle, H. 2015, in Proc. of Machine Learning Research, 37, Proc. of the 32nd Int. Conf. on Machine Learning, ed. F. Bach & D. Blei (Lille: PMLR), 881
Gillet, N., Mesinger, A., Greig, B., Liu, A., & Ucci, G. 2019, MNRAS, 484, 282
Giri, S. K., & Mellema, G. 2021, MNRAS, 505, 1863
Gleser, L., Nusser, A., Ciardi, B., & Desjacques, V. 2006, MNRAS, 370, 1329
Gong, Y., Cooray, A., Silva, M., et al. 2012, ApJ, 745, 49
Gong, Y., Cooray, A., Silva, M. B., Santos, M. G., & Lubin, P. 2011, ApJL, 728, L46
Goodfellow, I., Bengio, Y., & Courville, A. 2016, Deep Learning (Cambridge, MA: MIT Press), www.deeplearningbook.org
Gorce, A., Hutter, A., & Pritchard, J. R. 2021, A&A, 653, A58
Greig, B., & Mesinger, A. 2015, MNRAS, 449, 4246
Greig, B., & Mesinger, A. 2017, MNRAS, 472, 2651
Greig, B., & Mesinger, A. 2018, MNRAS, 477, 3217
Hahn, C., Vakili, M., Walsh, K., et al. 2017, MNRAS, 469, 2791
Harker, G. J. A., Zaroubi, S., Thomas, R. M., et al. 2009, MNRAS, 393, 1449
Harris, C. R., Millman, K. J., van der Walt, S. J., et al. 2020, Natur, 585, 357
Hassan, S., Andrianomena, S., & Doughty, C. 2020, MNRAS, 494, 5761
Hassan, S., Liu, A., Kohn, S., & La Plante, P. 2019, MNRAS, 483, 2524
Ho, M., Farahi, A., Rau, M. M., & Trac, H. 2021, ApJ, 908, 204
Hoffmann, K., Mao, Y., Xu, J., Mo, H., & Wandelt, B. D. 2019, MNRAS, 487, 3050
Hortúa, H. J., Malago, L., & Volpi, R. 2020a, MLST, 1, 035014
Hortúa, H. J., Volpi, R., & Malagò, L. 2020b, arXiv:2005.02299
Hortúa, H. J., Volpi, R., Marinelli, D., & Malagò, L. 2020c, PhRvD, 102, 103509
Hunter, J. D. 2007, CSE, 9, 90
Hutter, A., Dayal, P., Yepes, G., et al. 2021, MNRAS, 503, 3698
Hutter, A., Watkinson, C. A., Seiler, J., et al. 2020, MNRAS, 492, 653
Intema, H. T., Jagannathan, P., Mooley, K. P., & Frail, D. A. 2017, A&A, 598, A78
Ioffe, S., & Szegedy, C. 2015, in Proc. of the 32nd Int. Conf. on Machine Learning (Lille: PMLR), 448
Ishida, E. E. O., Vitenti, S. D. P., Penna-Lima, M., et al. 2015, A&C, 13, 1
Jasche, J., & Wandelt, B. D. 2013, MNRAS, 432, 894
Jennings, W. D., Watkinson, C. A., & Abdalla, F. B. 2020, MNRAS, 498, 4518
Jennings, W. D., Watkinson, C. A., Abdalla, F. B., & McEwen, J. D. 2019, MNRAS, 483, 2907
Jensen, H., Zackrisson, E., Pelckmans, K., et al. 2016, ApJ, 827, 5
Kacprzak, T., Herbel, J., Amara, A., & Réfrégier, A. 2018, JCAP, 2018, 042
Kamran, M., Ghara, R., Majumdar, S., et al. 2021, MNRAS, 502, 3800
Kapahtia, A., Chingangbam, P., Ghara, R., Appleby, S., & Choudhury, T. R. 2021, JCAP, 2021, 026
Kern, N. S., & Liu, A. 2021, MNRAS, 501, 1463
Kern, N. S., Liu, A., Parsons, A. R., Mesinger, A., & Greig, B. 2017, ApJ, 848, 23
Kodi Ramanah, D., Wojtak, R., & Arendse, N. 2021, MNRAS, 501, 4080
La Plante, P., Lidz, A., Aguirre, J., & Kohn, S. 2020, ApJ, 899, 40
Lewis, A. 2019, arXiv:1910.13970
Li, W., Xu, H., Ma, Z., et al. 2019, MNRAS, 485, 2628
Lidz, A., Furlanetto, S. R., Oh, S. P., et al. 2011, ApJ, 741, 70

Lin, C.-A., & Kilbinger, M. 2015, A&A, 583, A70
List, F., & Lewis, G. F. 2020, MNRAS, 493, 5913
Lueckmann, J.-M., Bassetto, G., Karaletsos, T., & Macke, J. H. 2018, in Proc. of The 1st Symp. on Advances in Approximate Bayesian Inference, ed. F. Ruiz et al. (Montreal: PMLR), 32
Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., et al. 2017, in Advances in Neural Information Processing 30 (NIPS 2017), ed. I. Guyon et al., 30 (Red Hook, NY: Curran Associates Inc.), 1290
Ma, Q., Helgason, K., Komatsu, E., Ciardi, B., & Ferrara, A. 2018, MNRAS, 476, 4025
Majumdar, S., Kamran, M., Pritchard, J. R., et al. 2020, MNRAS, 499, 5090
Majumdar, S., Pritchard, J. R., Mondal, R., et al. 2018, MNRAS, 476, 4007
McGreer, I. D., Mesinger, A., & D'Odorico, V. 2015, MNRAS, 447, 499
McKay, M. D., Beckman, R. J., & Conover, W. J. 1979, Technometrics, 21, 239
Mellema, G., Koopmans, L. V. E., Abdalla, F. A., et al. 2013, ExA, 36, 235
Mertens, F. G., Mevius, M., Koopmans, L. V. E., et al. 2020, MNRAS, 493, 1662
Mesinger, A., & Furlanetto, S. 2007, ApJ, 669, 663
Mesinger, A., Furlanetto, S., & Cen, R. 2011, MNRAS, 411, 955
Paciga, G., Albert, J. G., Bandura, K., et al. 2013, MNRAS, 433, 639
Papamakarios, G., & Murray, I. 2016, in Proc. of the 30th Int. Conf. on Neural Information Processing Systems, ed. D. Lee et al. (Red Hook, NY: Curran Associates), 1036
Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. 2021, JMLR, 22, 1
Papamakarios, G., Pavlakou, T., & Murray, I. 2017, in Proc. of the 31st Int. Conf. on Neural Information Processing Systems (Red Hook, NY: Curran Associates Inc.)
Park, J., Mesinger, A., Greig, B., & Gillet, N. 2019, MNRAS, 484, 933
Parsons, A. R., Backer, D. C., Foster, G. S., et al. 2010, AJ, 139, 1468
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, JMLR, 12, 2825
Perreault Levasseur, L., Hezaveh, Y. D., & Wechsler, R. H. 2017, ApJL, 850, L7
Planck Collaboration, Adam, R., Aghanim, N., et al. 2016, A&A, 596, A108
Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2016, A&A, 594, A13
Planck Collaboration, Aghanim, N., Akrami, Y., et al. 2020, A&A, 641, A6
Pober, J. C., Ali, Z. S., Parsons, A. R., et al. 2015, ApJ, 809, 62
Ramanah, D. K., Wojtak, R., Ansari, Z., Gall, C., & Hjorth, J. 2020, MNRAS, 499, 1985
Robertson, B. E., Ellis, R. S., Furlanetto, S. R., & Dunlop, J. S. 2015, ApJL, 802, L19
Robin, A. C., Reylé, C., Fliri, J., et al. 2014, A&A, 569, A13
Santurkar, S., Tsipras, D., Ilyas, A., & Mądry, A. 2018, in Nin Proc. of the 32nd Int. Conf. on Neural Information Processing Systems, NIPS'18 (Red Hook, NY: Curran Associates Inc.), 2488
Saxena, A., Majumdar, S., Kamran, M., & Viel, M. 2020, MNRAS, 497, 2941
Schafer, C. M., & Freeman, P. E. 2012, in Statistical Challenges in Modern, 3 ed. E. D. Feigelson & G. J. Babu (New York: Springer New York)
Schmit, C. J., & Pritchard, J. R. 2018, MNRAS, 475, 1213
Selvaraju, R. R., Cogswell, M., Das, A., et al. 2017, in IEEE Int. Conf. on Computer Vision (ICCV) (Piscataway, NJ: IEEE), 618
Shimabukuro, H., & Semelin, B. 2017, MNRAS, 468, 3869
Shimabukuro, H., Yoshiura, S., Takahashi, K., Yokoyama, S., & Ichiki, K. 2015, MNRAS, 451, 467
Shimabukuro, H., Yoshiura, S., Takahashi, K., Yokoyama, S., & Ichiki, K. 2016, MNRAS, 458, 3003
Shimabukuro, H., Yoshiura, S., Takahashi, K., Yokoyama, S., & Ichiki, K. 2017, MNRAS, 468, 1542
Simonyan, K., Vedaldi, A., & Zisserman, A. 2014, arXiv:1312.6034
Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. 2014, arXiv:1412.6806
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. 2014, JMLR, 15, 1929
Tieleman, T., & Hinton, G. 2012, COURSERA: Neural networks for machine learning, 4, 26
Tingay, S. J., Goeke, R., Bowman, J. D., et al. 2013, PASA, 30, e007
Trott, C. M., Jordan, C. H., Midgley, S., et al. 2020, MNRAS, 493, 4711
van Haarlem, M. P., Wise, M. W., Gunst, A. W., et al. 2013, A&A, 556, A2
Van Rossum, G., & Drake, F. L. 2009, Python 3 Reference Manual (Scotts Valley,CA: CreateSpace)
Van Rossum, G., & Drake, F. L., Jr. 1995, Python reference manual (Amsterdam: Centrum voor Wiskunde en Informatica Amsterdam)
Villanueva-Domingo, P., & Villaescusa-Navarro, F. 2021, ApJ, 907, 44
Virtanen, P., Gommers, R., Oliphant, T. E., et al. 2020, NatMe, 17, 261
Wagner-Carena, S., Park, J. W., Birrer, S., et al. 2021, ApJ, 909, 187
Watkinson, C. A., Greig, B., & Mesinger, A. 2022, MNRAS, 510, 3838
Weyant, A., Schafer, C., & Wood-Vasey, W. M. 2013, ApJ, 764, 116
Yoshiura, S., Shimabukuro, H., Takahashi, K., et al. 2015, MNRAS, 451, 266
Zeiler, M. D., & Fergus, R. 2014, European Conference on Computer Vision (Berlin: Springer), 818
Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. 2016, in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (Piscataway, NJ: IEEE), 2921
Zhou, M., Tan, J., & Mao, Y. 2021, ApJ, 909, 51