Deep learning simulations of the microwave sky

Dongwon Han,[1] Neelima Sehgal,[1] and Francisco Villaescusa-Navarro[2,3]
[1]*Physics and Astronomy Department, Stony Brook University, Stony Brook, New York 11794, USA*
[2]*Department of Astrophysical Sciences, Princeton University, 4 Ivy Lane, Princeton, New Jersey 08544, USA*
[3]*Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, New York, New York 10010, USA*

(Received 1 June 2021; accepted 4 November 2021; published 7 December 2021)

We present 500 high-resolution, full-sky millimeter-wave deep learning (DL) simulations that include lensed CMB maps and correlated foreground components. We find that these MillimeterDL simulations can reproduce a wide range of non-Gaussian summary statistics matching the input training simulations, while only being optimized to match the power spectra. The procedure we develop in this work enables the capability to mass produce independent full-sky realizations from a single expensive full-sky simulation, when ordinarily the latter would not provide enough training data. We also circumvent a common limitation of high-resolution DL simulations that they be confined to small sky areas, often due to memory or GPU issues; we do this by developing a "stitching" procedure that can recover the large-scale, high-order statistics and avoid discontinuities or repeated features in the maps. In addition, since our network takes as input a full-sky lensing convergence map, it can in principle take a full-sky lensing convergence map from any large-scale structure (LSS) simulation and generate the corresponding lensed CMB and correlated foreground components at millimeter wavelengths; this is especially useful in the current era of combining results from both CMB and LSS surveys, which require a common set of simulations.

DOI: 10.1103/PhysRevD.104.123521

I. INTRODUCTION

The cosmic microwave background (CMB) has been a cornerstone of modern precision cosmology. The study of the primordial anisotropy imprinted on the CMB at the surface of last scattering has shed light on the physics of the early Universe. A rich set of secondary anisotropies induced by intervening large scale structures (LSS) and complex astrophysical phenomena has revealed a picture of the evolving Universe. As the sensitivity of CMB experiments has improved, more information has been obtained from these features. A few recent examples include the constraints on cosmological parameters from CMB power spectra [1–5], the measurement of CMB lensing auto spectra [6–8], and the study of kinematic and thermal Sunyaev-Zel'dovich effects [9–12].

Current CMB data analyses rely heavily on simulations to (1) model the underlying physical processes, (2) verify the data analysis pipeline, (3) investigate potential biases, and (4) generate covariance matrices used in likelihood analyses. As a result, there have been efforts to improve simulations of the microwave sky [13–22]. Current and future high-resolution CMB experiments, such as the Atacama Cosmology Telescope (ACT) [23,24], the South Pole Telescope (SPT) [25], the Simons Observatory (SO) [26], CMB-S4 [27], and CMB-HD [28,29] will observe large sky areas at arcminute resolution or better with unprecedented sensitivity. Analyzing the data from these observations will need many realizations of realistic simulations of comparable resolution and sky coverage. In particular, as the sensitivity of CMB experiments improves, simulations capturing the correlations between foreground components and non-Gaussian features will become ever more important.

Simulations of the millimeter-wave sky can be divided into several components: (1) primordial CMB maps, (2) extragalactic foregrounds, and (3) Galactic foregrounds. The procedure to generate simulations of primordial CMB maps is well established and computationally efficient at arcminute resolution [30–34]. The extragalactic foreground components include the lensing convergence field (κ), the kinetic and thermal Sunyaev-Zel'dovich effects (kSZ and tSZ), the cosmic infrared background (CIB), and radio galaxies (radio). There are a few extragalactic foreground simulations publicly available that include all the above non-Gaussian, correlated foregrounds at high-resolution over a large fraction of the sky [19,22]. These simulations are generated by first making three-dimensional realizations of cosmological features, and then projecting them along the line-of-sight to make two-dimensional maps. To a large extent, these existing simulations can reproduce the statistics of the latest CMB observations. However, there are only a limited number of realizations available due to the high computational cost of generating initial three-dimensional

realizations. As a result, many CMB analyses model extragalactic foregrounds as Gaussian random fields matched to the observed foreground power spectra (e.g., [4–6]).

There have also been a number of efforts to simulate Galactic foregrounds in both intensity and polarization [35–43]. Modeling the small-scale features of Galactic foregrounds is particularly challenging given the current lack of observations on comparable scales in millimeter wavebands, though there is progress in this direction [44]. In this work, we focus on generating simulations including the primordial CMB and extragalactic foreground components, and leave inclusion of Galactic foregrounds to future work.

Recently, Machine Learning has found a broad application in cosmology; parameter estimation [45–52], foreground cleaning [42,43,53,54], feature extraction [55–58], inpainting [59–61], filtering [62], and modeling of physical processes [43,63–69]. Deep learning (DL) is a subset of machine learning that fits *multiple* nonlinear functions to the input data using neural networks. In particular, generative DL models aim at sampling the manifold (which can be high-dimensional) where the input data lives, so that new points in that manifold are associated with new complex data with the same statistical properties as the desired input data. Recently, there have been many approaches in this direction [42–44,70–81]. However, these methods are not yet widely used in CMB data analyses due to their limited sky-coverage or summary statistics mismatch.

In this work, we generate for the first time, many independent full-sky millimeter-wave DL simulations that include all correlated extragalactic foreground components appropriate for CMB analyses. We find that our MillimeterDL (hereafter mmDL) simulations are able to reproduce well the non-Gaussian statistics of the original input maps. Several recent works have focused on using various DL techniques to generate two-dimensional projected microwave fields from three-dimensional N-body simulations [70,71,73–78,80,81]; in this work, we use two-dimensional projected simulations at 148 GHz described in [19] (hereafter S10) as the input data to train our DL algorithm, as opposed to three-dimensional N-body simulations. This is suitable for generating maps for a CMB imaging survey, since most of the relevant information for CMB data analyses is contained in the two-dimensional projected maps. The advantage of our approach is that DL training in two dimensions is computationally more efficient than that in three dimensions; therefore, for a given computing resource, one can train on a larger footprint at higher-resolution.

In particular, we use DL methods to generate 500 realizations of full-sky millimeter-wave simulations that match the statistics of the S10 simulations. Each of these 500 simulation realizations includes:

(1) a full-sky simulation at half-arcminute resolution at six different frequencies (30, 90, 148, 219, 277, and 350 GHz).
(2) the lensed CMB in both temperature and polarization.
(3) extragalactic foreground components appropriately correlated with each other, including:
   (i) the lensing convergence map (κ).
   (ii) the kinetic Sunyaev-Zel'dovich effect (kSZ).
   (iii) the thermal Sunyaev-Zel'dovich effect (tSZ).
   (iv) the cosmic infrared background (CIB).
   (v) the radio galaxies (radio).
(4) non-Gaussian information that reproduces statistics such as the one-point function, bispectra, and trispectra.

These mmDL simulations are trained using the cosmology and extragalactic model described in S10 [19]. However, the method presented here is general and can be applied to reproduce other two-dimensional simulations with minor modifications. In Sec. III, we present our procedure to generate the mmDL simulations. The comparison between the statistical properties of the mmDL and the S10 simulations is shown in Sec. IV. We discuss potential applications and the data products released in Sec. VI. These mmDL simulations are publicly available in both HEALPix[1] and CAR formats on the Legacy Archive for Microwave Background Data Analysis (LAMDA)[2] and the National Energy Research Scientific Computing Center (NERSC) cluster.[3] We also provide the code to produce additional realizations at [82].

II. DL PRIMARY INPUT DATA

We generate our *Primary Input Data* from the S10 simulations, which include κ, kSZ, tSZ, CIB and radio [19] extragalactic foreground components. The S10 simulations were generated by post-processing the output of a three-dimensional simulation, which has *WMAP5* cosmology [83]. The S10 simulations have correlated foreground components, and have been shown to have non-Gaussian statistics that match observations [84–88]. The S10 simulations and source catalogs are publicly available on NASA's LAMBDA website (see Footnote 2). The simulations are saved in HEALPix format [89] with Nside $= 8192$, and have units of Jy/sr. The S10 κ map has Nside $= 4096$, and is dimensionless.

We describe our procedure to prepare the mmDL primary input data below. Our procedure includes (1) pre-processing the original S10 148 GHz simulations, 2)) dividing the pre-processed S10 simulations into small patches, and (3) applying a normalization to the datasets to stabilize and speed up the training of a neural network.

[1] http://healpix.sourceforge.net.
[2] https://lambda.gsfc.nasa.gov/simulation/tb_sim_ov.cfm.
[3] https://crd.lbl.gov/departments/computational-science/c3/c3-research/cosmic-microwave-background/cmb-data-analysis-at-nersc/.

A. Preprocessing of the S10 Simulations

Below we detail the steps in our procedure to preprocess the original HEALPix S10 simulations for the network training.

(i) Following [26,88], we scale the flux of CIB sources and the tSZ map by a factor of 0.75 to make the S10 simulations more closely match recent measurements from ACT, Planck and SPT [90–94].

(ii) We convert the units of the maps from Jy/sr to μK, except for the κ map which is dimensionless.

(iii) We reproject the κ, tSZ, and kSZ foreground components of the S10 simulations from HEALPix pixelization to plate carrée (CAR) pixelization at half arcminute resolution using the public *pixell*[4] and *libsharp*[5] [95] libraries. Since CAR maps intrinsically consist of a grid of rectangular pixels, there is a natural way to extract two-dimensional submaps; this makes it simpler to train DL networks and apply two-dimensional Fourier transform operations. On the other hand, CAR pixels are increasingly oversampled farther away from the equator (the image is stretched), which we discuss below. The reprojection is done by converting the original HEALPix maps to spherical harmonic a_{lm} up to an $\ell_{\rm max}$ of 10,000. These a_{lm} are then reprojected onto full-sky CAR pixelized maps.

(iv) For the CIB and radio foreground components, we place sources directly from the S10 catalogs into full-sky CAR pixelized maps to avoid spectral leakage ("ringing") around pointlike sources when manipulating them in harmonic space. We apply a flux-cut of 7 mJy at 148 GHz to both CIB and radio CAR maps in order to match current and future CMB analyses that typically detect and remove bright sources.

(v) Since CAR maps require the same number of pixels at each declination, the image appears stretched at the poles. To prevent the network from learning nonphysical features induced by this stretching, we use only the region of the simulations near the equator to train the network (from -10° to 10° in declination). However, in order to make use of all the sky area available in the S10 simulations, we rotate the full S10 map many times, redefining the equator of the map each time. In particular, we rotate each full-sky map by 15 degrees in ψ and 20 degrees in θ, where ψ and θ are the first two Euler angles. The successive rotations in ψ and θ described above generate 25 unique cylindrical strips around each new equator, which strips in CAR projection have minimal image distortion compared to the sphere. For κ, tSZ, and kSZ, the rotations are done in spherical harmonic space. For CIB and radio, we simply remap source positions for given rotation angles.

B. Division of full sky simulations into smaller patches

To train a network, we need many training samples. Since we start with a single realization of the full sky S10 simulation, we increase the number of samples by dividing this simulation into smaller overlapping patches. Below we detail our steps.

(i) We randomly pull out patches of 128×128 pixels (roughly $1^\circ \times 1^\circ$ patches) between the declination of -10° and 10°. We cut out patches in this way from our 25 cylindrical strips (described above) for κ, tSZ, kSZ, CIB and radio maps. Thus each patch has five foreground components totaling an array of size $5 \times 128 \times 128$. We repeat this step until we collect 30,000 validation samples (to pick initial parameters for the training), 200,000 training samples, and 30,000 test samples (to check the performance of the training). We ensure that we have an equal number of samples from the 25 rotated maps. We call the resulting dataset the primary input data. Note that these samples will necessarily have overlapping regions among them. We discuss our method to overcome this issue in Sec. III.

(ii) During the network training, we randomly flip a sample either vertically or horizontally, or both as a natural way to augment the training dataset.

C. Normalization to speed up and stabilize the training

We need to normalize the samples to stabilize and to speed up the neural network training. Since we have multiple foreground components in a single sample, the normalization needs to account for two issues. First, the pixel intensity probability density function (pixel PDF) of different foreground components follows different distributions. The pixel PDFs of tSZ, CIB, and radio roughly follow a Poisson distribution, whereas those of κ and kSZ are better described by a Gaussian distribution. Second, these pixel PDFs have varying dynamic ranges. For example, all the pixel values of the κ map fall between -1.5 and 1.5; on the other hand, the radio map has pixel values ranging from 0 to 852 μK. Ideally, we would like the pixel PDFs to follow a standard Gaussian distribution (i.e., with unit variance) for faster and more stabile network training. To address the shape of the pixel PDF, we first perform a pixel-wise natural log scaling adopted from [74], to the tSZ, CIB, and radio maps. In particular, we replace the value of each pixel, given by v, with

$$f(v) = \mathrm{sgn}(v) \ln\left[\frac{\mathrm{abs}(v)}{\sigma_v} + 1\right] \tag{1}$$

where $\mathrm{sgn}(v)$ is sign function, and σ_v the standard deviation of the pixel PDF of the input map. After this normalization,

[4] https://github.com/simonsobs/pixell.
[5] https://github.com/Libsharp/libsharp.

the pixel PDFs of the tSZ, CIB, and radio maps are closer to a Gaussian distribution.

To mitigate the dynamic range issue, we then standardize the maps using the standard score method (i.e., subtract the mean of all the pixels from each pixel value and then divide that by the variance of the map).[6]

$$s(v') = (v' - \mu_{v'})/\sigma_{v'} \tag{2}$$

Here, v' is the pixel value, and $\mu_{v'}$ and $\sigma_{v'}$ are the mean and standard deviation of the pixel values respectively.[7] Note that each foreground component is normalized independently from the others. The parameter values used in the normalization procedure are shown in Appendix A. Note that each of these normalization operations has a well-defined inverse operation. When we generate the final products, we undo these normalizations prior to the interpolation procedures discussed in Sec. III D.

III. METHOD

Generative adversarial networks (GANs) are a class of networks that will generate new data matching the statistics of the training dataset [96]. Fundamental to a GAN is a set of two neural networks, called the generator and discriminator. A generator makes new data, while a discriminator tries to distinguish between the real data and the generated data. By iteratively training a generator and a discriminator, a GAN reaches an equilibrium where the discriminator can no longer distinguish new data from training data. For more discussion about GANs, we refer the reader to [97,98].

GAN models can only generate random samples limited to the same footprint as the training data. Since our primary input data consists of $1^\circ \times 1^\circ$ patches, the random samples output by the GAN do not have the full information about modes on scales larger than 1 degree (roughly $\ell \leq 200$). This presents a challenge in stitching together the small patches to make a full-sky simulation with the correct large-scale fluctuations. Furthermore, there is no straightforward way to tile these random patches together to make a full-sky map without having discontinuities at the tile edges.

Instead, we train a conditional GAN network to predict the four extragalactic foreground components (tSZ, kSZ, CIB, and radio) from a κ map. Once this network training step is completed, we generate a full-sky Gaussian κ map, and convert it to the other four full-sky extragalactic components tile by tile. Since the input Gaussian κ maps is continuous, the resulting extragalactic foreground maps are also continuous. This image prediction approach also solves another major concern about using DL to generate new samples, which is mode collapse. Mode collapse is when the network predicts only a subset of all possible outcomes, instead of a continuous distribution. However, with our approach, since we provide a unique input full-sky κ map to the network for each full-sky realization the network outputs, we can ensure that the output maps are also unique.

A challenge of this predictive approach, which uses a conditional GAN model, is that it has a large number of parameters that need to be trained; thus this conditional GAN requires a much larger training dataset than simple GANs. Since our primary input data is relatively small and has over-lapping patches (see Sec. II for details), we first use the primary input data to train a simple GAN network (which is easier to train) to make a large sample of non over-lapping patches of κ, tSZ, kSZ, CIB and radio maps (intermediate product 1 in Fig. 1). This larger dataset is then used to train a conditional GAN (training step 2 in Fig. 1). After training step 2 is done, the network can now predict extragalactic foregrounds given a κ map. We input a unique full-sky Gaussian κ map into the network to generate the other extragalactic foreground components (tSZ, kSZ, CIB and radio) tile by tile (intermediate product 2 in Fig. 1). The extragalactic components predicted from the Gaussian κ map will be missing some non-Gaussian information due to the missing non-Gaussian information in the κ map; thus, we introduce a simpler version of the conditional GAN that can restore this missing non-Gaussian information, which we train on the primary input data plus intermediate product 2. Once this training is done, the network is equipped to go from unique input full-sky Gaussian κ maps and unlensed CMB T, Q, and U maps to a full-sky simulation including non-Gaussian extragalactic foregrounds, a non-Gaussian κ map, and the corresponding lensed CMB T, Q, and U maps (T, Q, and U are the temperature and two polarization fields of the CMB). The schematic of the overall procedure is shown in Fig. 1.

Since the extragalactic maps are statistically invariant under translation, a natural choice is to use a deep convolutional generative adversarial network (DCGAN) since it has convolution kernels that are also invariant under translation. In this work, we use three variations of the DCGAN: (1) Deep convolutional Wasserstein GAN with gradient penalty (DCWGAN-GP) to make intermediate product 1 (this is training step 1 discussed in Sec. III A), (2) Pix2Pix GAN (PIXGAN) to make to make intermediate product 2 (this is training step 2 discussed in Sec. III B), and (3) Variational autoencoder GAN (VAEGAN) to augment the missing non-Gaussian information (this is training step 3 discussed in Sec. III C).

A. Training step 1: Learning on small patches with Wasserstein GAN with gradient penalty (DCWGAN-GP)

The DCWGAN-GP model we use is summarized by the generator depicted in Fig. 2 and the discriminator described

[6]Another popular approach is to scale an image such that all the pixel values lie between -1 and 1. During our network training, we found that using the standard score method leads to more stable training compared to this rescaling.

[7]Note that we distinguish v and v' here because the pixel values of the tSZ, CIB, and radio maps are first scaled by Equation (1) (i.e., $p' = f(p)$), whereas $p' = p$ for the κ and kSZ maps.

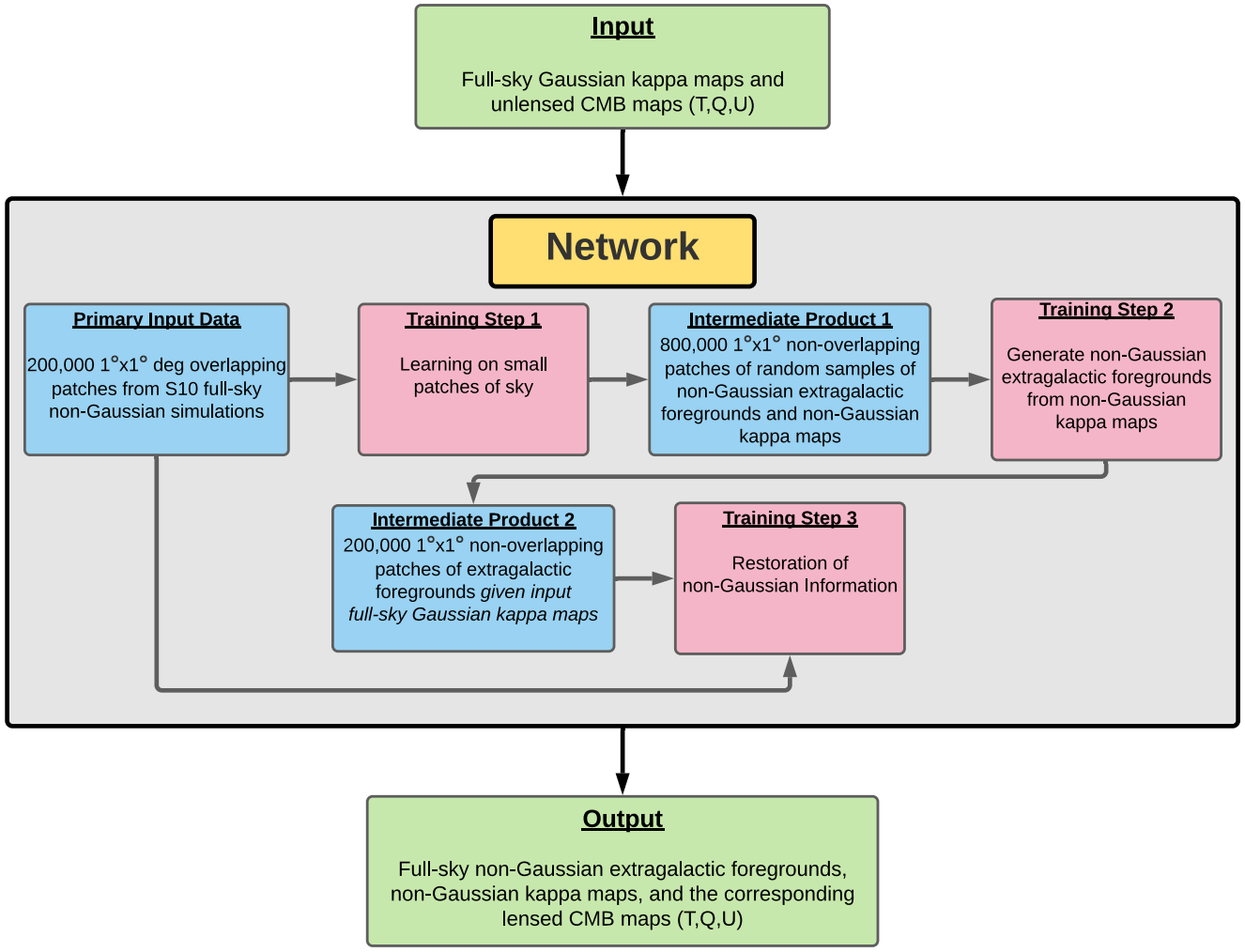

FIG. 1. Shown is a schematic of the overall procedure to train the network and generate the output maps from the input data. The network is represented by the shaded gray box in the center. The mmDL procedure starts with the Primary Input Data consisting of 200,000 $1^\circ \times 1^\circ$ overlapping patches cut out from the original S10 simulations. The pink boxes represent the three major training steps described in Sec. III. The blue boxes represent inputs into each training step. The dark grey arrows indicate the input and output of each step. Once the network training is completed, we feed full-sky Gaussian kappa maps and unlensed CMB maps (T, Q, U) into the network (top green box); given these inputs, the network generates the final output products (bottom green box), which are full-sky millimeter-wave simulations including lensed T, Q, and U maps, non-Gaussian kappa maps, and non-Gaussian extragalactic foregrounds correlated with the kappa map and each other.

in Table I. We take the *CosmoGAN* network architecture used in [72] as a starting point of our DCWGAN-GP network because it has been demonstrated to reproduce non-Gaussian statistics for a one-component map. From this starting point, we make the following adjustments. First, we switch the simple GAN loss function used in *CosmoGAN* with the Wasserstein loss function with gradient penalty introduced in [99]; the loss function is the statistic that all GANs optimize. We find that this change of the loss-function substantially improves the stability of our network against mode-collapsing (i.e., against the network missing critical features). To be consistent with the standard Wasserstein GAN (WGAN) architecture, we remove the batch normalization layers (i.e., the intermediate normalization steps within the GAN) from the discriminator to stabilize the loss function gradient, and replace the sigmoid function activation layer with a linear function activation layer, following [99]. In addition, we increase the depth of the convolution layers in both generators and discriminators from four to five, and we increase the latent-vector length (i.e., array size of input random numbers) from the original 64 to 256 in order to have a more expressive network. We also change the last activation layer in the generator model from $\tanh x$ to $a * \tanh b/a * x$, which we call a scaled tanh. Here a is chosen to be 15 to accommodate the dynamic range of the normalized pixel values we obtained using Eq. (1), while b is chosen to be 2 so that the gradient around the boundary values (i.e., -15 and 15) do not vanish too rapidly; we find the form of the scaled

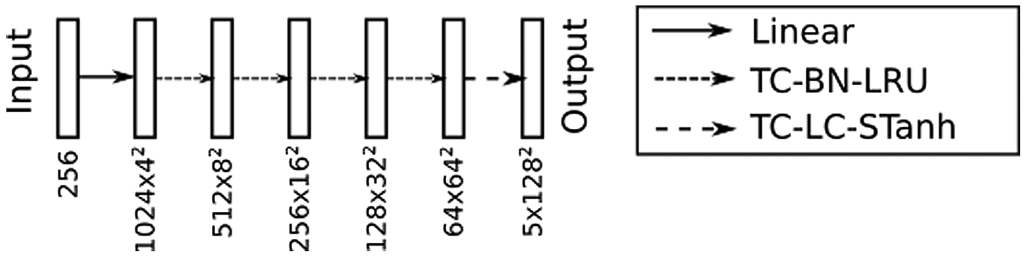

FIG. 2. Shown is a diagram of the DCWGAN-GP generator model described in Sec. III A and indicated in the schematic of Fig. 1 as training step 1. The input to this generator is a latent-vector consisting of 256 random numbers drawn from a Gaussian distribution with mean equal to zero and variance equal to one ($\mu = 0$, $\sigma = 1$). The output is one set of $1° \times 1°$ maps consisting of the five foreground components. The direction of data flow is indicated by arrows. A solid arrow is a fully connected linear layer. A short-dashed arrow, labeled TC-BN-LRU, represents a sequence of (1) a 4×4 two-dimensional transposed convolution, (2) a batch normalization, and (3) a leaky rectified linear unit activation function with $\alpha = 0.2$. Lastly, a long-dashed arrow, labeled TC-LC-STanh, is a sequence of (1) a 4×4 two-dimensional transposed convolution, (2) a linear channel, and (3) a scaled tanh activation function. The linear channel and scaled tanh are defined in Sec. III A. Note that a transposed convolution upsamples an image by a factor of two.

tanh function and its parameters by checking the training statistics when using the validation set of patches (see Sec. II B). We also add a custom linear layer that linearly rescales each extragalactic component independently so that the network has more freedom to rescale the components relative to each other. We call this custom layer the "linear channel" (LC). We find that using this LC helps the network to converge faster. We reduce the two-dimensional convolution layer kernel size (i.e., the width and the height of a convolution filter) from 5×5, used in *CosmoGAN*, to 4×4 to account for the difference in the input image size from 256×256 pixels to 128×128 pixels (the latter is for $1° \times 1°$ patches at 0.5 arcmin resolution). Lastly, we replace the rectified linear unit (ReLU) layers in the generator with the leaky rectified linear unit (Leaky ReLU) with $\alpha = 0.2$ to achieve better performance, as suggested by [100]. (For more discussion about various types of activation functions, we refer to [101]).

We train the DCWGAN-GP network with the primary input data using up to 100 epochs (i.e., the number of cycles through the entire training dataset used by the network during training). Following [99], we adopt a gradient penalty coefficient (λ) of 10 and the number of discriminator iterations per generator iteration ($n_{\rm critic}$) of 5. For gradient decent, we use the Adam optimizer [102] with a learning rate (lr) of 10^{-4} and $(\beta_1, \beta_2) = (0.5, 0.9)$. Throughout this training, we check not only the loss function values, but also the visual images, Minkowski functionals, power spectra, cross spectra, and pixel PDFs of the network outputs; we will discuss each metric in Sec. IV. If the network shows signs of overfitting or mode-collapsing, we restart the training either from the beginning or after reducing the learning rate by half (i.e., effectively reducing the step size). We terminate the training cycle once all the statistics considered are satisfactorily reproduced. We use the trained DCWGAN-GP network to produce *intermediate product 1* (800,000 $1° \times 1°$ patches of non-Gaussian κ maps and correlated non-Gaussian foregrounds) which we use in training step 2. Note that the samples in intermediate product 1 are no longer overlapping and are statistically independent of each other.

TABLE I. Summary of the discriminator model described in Sec. III A. We use the same discriminator model throughout all three training steps described in Sec. III. The first and second columns list the layers and the corresponding activation functions. The third and the fourth columns give the shapes of the output arrays and the number of trainable parameters in each layer. Here, N is the number of input components, i.e., $N = 5$ for DCWGAN-GP and PIXGAN and $N = 10$ for VAEGAN. Conv 4×4 indicates a two-dimensional convolution layer with a 4×4 kernel, and LReLU stands for a leaky rectified linear unit activation with $\alpha = 0.2$ (see Sec. III A for details). Note that Conv 4×4 downsamples an image by a factor of two, and increases the number of features to consider.

Layer	Activation function	Output shape of tensor	# of trainable parameters
Input map	N/A	$N \times 128 \times 128$	N/A
Conv 4×4	LReLU	$64 \times 64 \times 64$	5,184
Conv 4×4	LReLU	$128 \times 32 \times 32$	131,200
Conv 4×4	LReLU	$256 \times 16 \times 16$	524,544
Conv 4×4	LReLU	$512 \times 8 \times 8$	2,097,664
Conv 4×4	LReLU	$1024 \times 4 \times 4$	8,389,632
Linear	N/A	1	16,385
Total trainable parameters			11,164,609

B. Training step 2: Generate extragalactic foregrounds from kappa map with Pix2Pix GAN (PIXGAN)

The PIXGAN model we use is summarized by the generator depicted in Fig. 3 and the discriminator described in Table I. A PIXGAN can convert input images to other images by implementing a U-NET generator [103,104]. We use the PIXGAN network to convert a given κ map to the other four non-Gaussian extragalactic foregrounds consisting of tSZ, kSZ, CIB, and radio components. We start with the original PIXGAN architecture presented in [104], and make the following modifications. We first replace the tanh

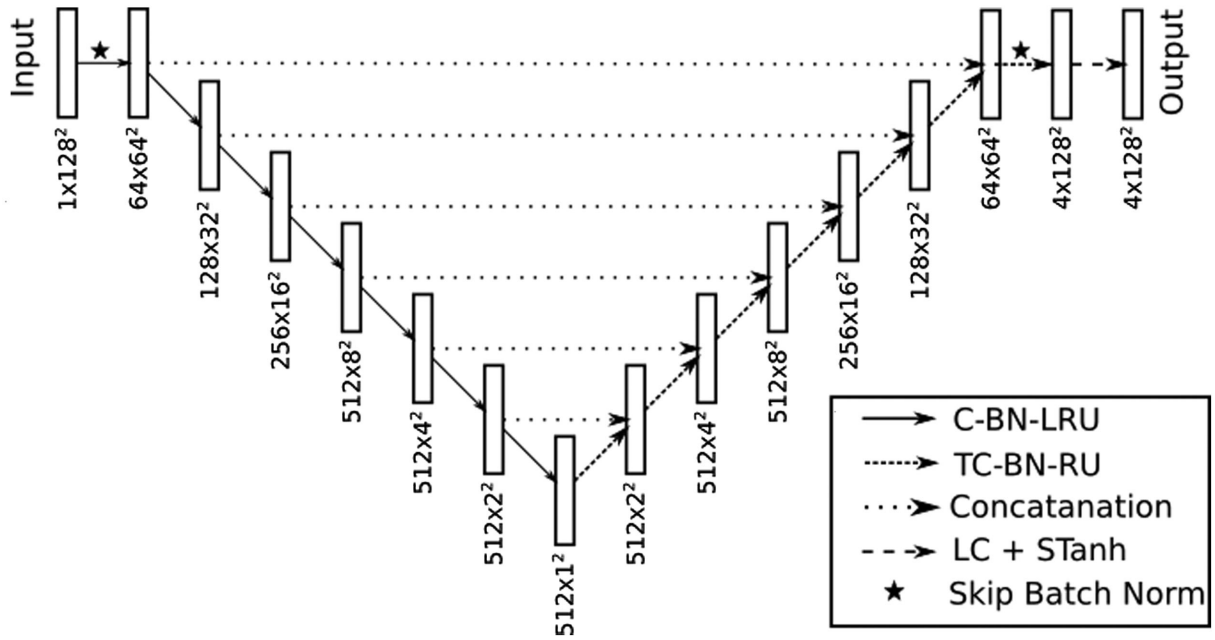

FIG. 3. Shown is a diagram of the UNET generator model described in Sec. III B and indicated in the schematic of Fig. 1 as training step 2. The generator takes as input a $1° \times 1°$ κ map, and it outputs a set of $1° \times 1°$ extragalactic foreground maps consisting of tSZ, kSZ, CIB, and radio components. The direction of the data flow is indicated by the arrows. A solid arrow, labeled C-BN-LRU, represents a sequence of (1) a 4×4 two-dimensional convolution, (2) a batch normalization, and (3) a leaky rectified linear unit activation function. A short-dashed arrow, labeled TC-BN-RU, represents a sequence of (1) a 4×4 two-dimensional transposed convolution, (2) a batch normalization, and (3) a rectified linear unit activation function. A two-dimensional convolution downgrades an image by a factor of two, while a transposed convolution upsamples an image by the same factor. A dotted arrow is a concatenation operation, which concatenates the output of an originating box to the output of a destination box. A long-dashed arrow, labeled LC + STanh, on the top right represents a sequence of a linear channel and a scaled tanh activation function. Linear channel and scaled tanh are defined in Sec. III A. Deviating from our notation, we do not apply a batch normalization to the first and the last convolution layers, indicated by a star over a line.

activation layer in the generator with a sequence of a linear channel and a scaled tanh; the latter two are defined in III A. Since we terminate the training well before the network shows signs of overfitting, we also remove the drop-out layers from the generator to speed up the training. Lastly, instead of using the PatchGAN discriminator as in [104], we use the same DCWGAN-GP discriminator model described in Sec. III A, and summarized in Table I.

We use intermediate product 1 (800,000 $1° \times 1°$ non-overlapping patches of non-Gaussian κ, tSZ, kSZ, CIB and radio maps) generated by training step 1 to train the PIXGAN network.[8] We use the same optimizer setup as in training step 1. During the training process, we find that the PIXGAN network tends to overestimate the number of massive tSZ clusters, which results in an excess of tSZ power at large scales. To mitigate this effect, we add a linear loss term to the DCWGAN-GP generator loss function evaluation defined as the second term in the right side of the equation below.

$$\mathcal{L}_G^{\mathrm{PIXGAN}} = \mathcal{L}_G^{\mathrm{DCWGAN-GP}} + \gamma \sum_i |\sigma^2(S^i_{\mathrm{network}}) - \sigma^2(S^i_{\mathrm{input}})| \qquad (3)$$

where $\mathcal{L}_G^{\mathrm{DCWGAN-GP}}$ is the standard DCWGAN-GP generator loss function defined in [99], and $\sigma^2(S^i)$ is the total pixel variance of the ith foreground component in a sample. Since the sum of power spectra values across multipoles is the equal to the total pixel variance of the map (a corollary of Parseval's theorem), it follows that penalizing the difference in the total pixel variance is equivalent to penalizing the mismatch in the summed power spectra. We train the model with $\gamma = 100$, and find that it helps to regulate the excess power in the tSZ map.

We train the PIXGAN network using up to 10 epochs.[9] We validate the network every 200,000 training samples (four times per epoch) with the updated loss function in Eq. (3), checking the same metrics as in Sec. III A. We intentionally terminate early the training of the PIXGAN network for two reasons. First, once the PIXGAN network

[8]We still use the test and the validation datasets from our primary input data to fine-tune the network parameters and to validate the network outputs.

[9]Note that each epoch here has fours times more samples than the epochs in Sec. III A.

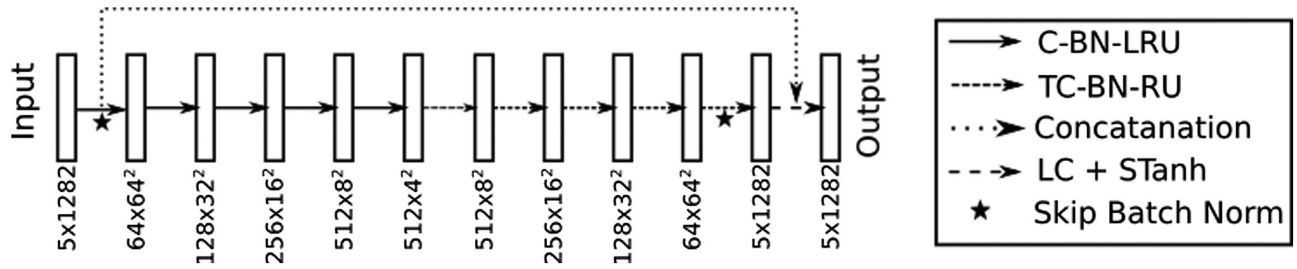

FIG. 4. Shown is a diagram of the VAEGAN generator model described in Sec. III C and indicated in the schematic of Fig. 1 as training step 3. The input to this generator is a set of $1° \times 1°$ *Gaussian* κ maps and their corresponding extragalactic foreground maps from the PIXGAN network in Fig. 3 (i.e., intermediate product 2). The output are a set of non-Gaussian κ maps and their corresponding non-Gaussian extragalactic foreground maps. The direction of the data flow is indicated by arrows. A solid arrow, labeled C-BN-LRU, represents a sequence of (1) a 4×4 two-dimensional convolution, (2) a batch normalization, and (3) a leaky rectified linear unit activation function. A short-dashed arrow, labeled TC-BN-RU, represents a sequence of (1) a 4×4 two-dimensional transposed convolution, (2) a batch normalization, and (3) a rectified linear unit activation function. The two-dimensional convolution downgrades the image by a factor of two, while the transposed convolution upsamples the image by the same factor. Shown as a dotted arrow is a concatenation operation between the input intermediate product 2 and the VAEGAN network output, which focuses the network on the residual difference between the two; note that we apply the inverse of scaled tanh to this input prior to the concatenation. A long-dashed arrow, labeled LC + STanh, represents a sequence of a linear channel and a scaled tanh activation function; they are described in Sec. III A. Deviating from our notation, we do not apply a batch normalization to the first and the last convolutions layers, indicated by a star below the arrow.

starts to overfit, the quality of the output degrades quickly. We find it more conservative to undertrain the PIXGAN network rather than to risk over-fitting. Second, we find that the VAEGAN network described in the next training step (training step 3) can absorb some imperfection in the simulations and correct for them. Therefore, the output from the PIXGAN network can have minor defects without affecting the quality of the final products. We terminate the training once we see signs of over-fitting; these signs are (1) minimal improvement of the loss function over the training samples, (2) large fluctuations of the other metrics with respect to the S10 simulations, and (3) images missing significant features. At this point, we choose the snapshot of the network that reproduces the summary statistics the best. In particular, for the power spectra, we check that the PIXGAN network can reproduce the shape of each foreground component power spectra up to a constant multiplication factor.[10] When this network snapshot reproduces the shape of the power spectra to within 10% for $\ell \in (2000, 8000)$, and the cross spectra and Minkowski functionals also show similar agreement, we consider the training complete. Otherwise, we repeat the training procedure until we reach this criteria. After this training step is done, we feed in several full-sky Gaussian κ maps through the trained PIXGAN network, producing 200,000 $1° \times 1°$ nonoverlapping patches of extragalactic foregrounds consisting of tSZ, kSZ, CIB, and radio components; we call this *intermediate product 2*. The full-sky Gaussian κ maps are generated using *pixell*; we use the power spectra of the S10 kappa map as the input spectra to these Gaussian kappa simulations.

[10]Note that the next network (training step 3) can rescale each foreground component by a constant factor as described in Sec. III C.

C. Training step 3: Restoration of non-Gaussian information with variational autoencoder GAN (VAEGAN)

The VAEGAN model we use is summarized by the generator depicted in Fig. 4 and the discriminator described in Table I. Neural networks can be used to improve existing simulations, either by improving the resolution or by adding missing small-scale information [44,105]. In this work, we adopt the variational autoencoder GAN (VAEGAN) in order to add back missing non-Gaussian information [106]. Our VAEGAN generator architecture is similar to the U-NET generator architecture discussed in Sec. III B except for the following. Unlike the PIXGAN, the VAEGAN generator does not have skip-connections for each layer (i.e., there are no array concatenations for each layer like as shown by the dotted lines in Fig. 3), except that we add back input maps from intermediate product 2 at the very end (see dotted line in Fig. 4) as done in [80]. This addition at the end allows the VAEGAN network to focus just on the missing non-Gaussian information (i.e., to focus on the residual difference between the network product of training step 3 and intermediate product 2). Another difference between the PIXGAN and VAEGAN generators are that the latter has fewer trainable parameters, which makes it easier to train; thus we only need 200,000 samples, as opposed to 800,000, to train it. Regarding the discriminator, we use the same DCWGAN-GP discriminator (i.e., the same Wasserstein loss function with the gradient penalty) discussed in Sec. III A, and summarized in Table I.

We use as input both the Primary Input Data and intermediate product 2 to train the VAEGAN network. The output of the VAEGAN network is the restoration of the missing non-Gaussian information in intermediate product 2. The VAEGAN network does this by comparing the original intermediate product 2 to the S10 simulations and learning the difference. To do this we use the same loss function and training setup as in training step 2. Since the output from training step 3 is our final product, except for a few corrections described later in Sec. III D, we further fine-tune the training step 3 training procedure to achieve optimal results. In particular, we revisit the issue of overestimation of massive tSZ clusters that we described in Sec. III B. In order to further reduce the chance of overestimation, we make the following adjustments to our training procedure: (1) first, we sort the training samples by the total variance of the tSZ component, (2) then we start training with the bottom eighty percent of samples in terms of the total tSZ variance; (3) Once the training is stabilized, we restart the training with the full set of training samples. We closely monitor the tSZ statistics along with the loss function and other summary statistics. We terminate the training when all the statistics considered are within 5% for $\ell \in (2000, 8000)$.

D. Post-processing the network simulations

As discussed in Sec. III, we convert a full-sky Gaussian κ map to other foreground components tile by tile, where each tile is a $1° \times 1°$ flat patch of sky. Here, we describe our method to divide full-sky κ maps into tiles and to reproject the processed tiles back on to a full-sky map. When we sample a $1° \times 1°$ patch from a full-sky κ map, we rotate that patch to the celestial equator and then cut out a tile of size 128×128 pixels with a 20-pixel overlap region between each tile as shown in Fig. 15 in Appendix B. This is similar to how we cut the original S10 simulations as discussed in Sec. II B. This results in a grid of tiles with 200 tiles along a longitudinal direction and 400 tiles along a latitude direction.[11] When we reproject the tiles back on to a full-sky map, we interpolate them back to the correct location on the sphere according to their coordinates. Here, we take advantage of the fact that CAR pixels in a full-sky map get stretched only along the latitude direction. As a result, the pixels in the longitudinal direction can be remapped without any interpolation. This effectively reduces the dimensions of the interpolation from two to one, which lowers the overall computational cost.

We use two different interpolation methods depending on which foreground component we would like to reproject. We use a linear interpolation to reproject the κ, tSZ, and kSZ foreground components because they are continuous fields.[12] To reproject the CIB and radio galaxies, we use the nearest neighbor method (i.e., each CAR pixel in the full-sky map obtains a value from the nearest pixel in the flat-sky tile). We use this method because linear interpolation can "blur" sharp variations in amplitude, which is not ideal for interpolating discrete point sources. We remove any superfluous negative flux sources before reprojecting point source tiles back onto the full-sky. Since we initially sample the full-sky κ map with overlapping regions, one pixel in the full-sky map can be mapped onto more than one flat-sky tile. Therefore, we precompute the number of mapped tiles for each full-sky map pixel and construct an effective "hit-counts" map. If more than one tile corresponds to a single full-sky pixel, then we add the tile pixel values and divide by the hit-counts map to obtain an average. In addition, we apodize each tile with a cosine apodization mask prior to the full-sky reprojection as done in [44] to suppress any discontinuity around the edges; this is done by weighting the overlapping regions by the apodization mask to downweight pixels closer to the edges of a given tile.[13] Lastly, following Sec. IV. 3 in [107], we compute two-dimensional transfer functions to correct for the reprojection effects for the κ, tSZ, and kSZ maps. Note that we do not apply two-dimensional transfer functions to the CIB and radio maps to avoid spectral leakage when shifting to Fourier space.

We find that the raw full-sky outputs from the network can reproduce the statistics of the S10 simulations overall quite well. However, we do find some small mismatches in the summary statistics between the network and the S10 simulations; for example, for the power spectra in the range $\ell \in (2000, 8000)$, the agreement is better than 3%, 10%, 0.5%, 17%, and 0.3% for κ, kSZ, tSZ, CIB and radio respectively, however, it is not perfect. Therefore, we make a few corrections to the network simulations to match the S10 simulations more closely. Here, we describe the corrections we apply. First, to correct for the small mismatch in the power spectra, we apply one-dimensional transfer functions to the κ, kSZ, and tSZ maps. These transfer functions are computed as the square root of the ratio of the average of 10 network power spectra to the S10 power spectra. We compute the transfer functions at each

[11]Around the poles of the sphere (i.e., the first and the last longitudinal rows of the grid), there is not enough physical area to sample a $1° \times 1°$ tile. Hence, we take the second and the second-to-the-last rows of the grid, mirror them longitudinally, and replace the first and the last rows of the grid with these mirrored rows. This ensures that we have continuous physical tiles around the poles at the cost of them not being unique. We find that this pole correction does not affect the overall statistics of the network simulations.

[12]We tried both linear interpolation and popular cubic spline methods, and found no noticeable difference in the resulting summary statistics. Thus, we chose to use the linear interpolation method which is more computationally efficient.

[13]We set the pixel values at the outermost edges of the apodization mask to the values of pixels one pixel away inside the map to avoid losing information.

multipole, and apply to these transfer functions a one-dimensional Gaussian filter with $\sigma_\ell = 10$ to smooth them out. We then convert these to isotropic two-dimensional transfer functions, and apply them to the spherical harmonics of the full-sky map.

We find larger deviations in the power spectrum between the network and the S10 simulations than specified above at larger ($\ell \leq 2000$) and smaller scales ($\ell \geq 8000$). The large scale deviation is due to the limited size of the sample patches as discussed in Sec. III. For the small scales, we find a roll-off in the power spectra starting at $\ell = 8000$. We find that this roll-off starts at larger scales ($\ell \leq 8000$) if we train the network with lower resolution simulations (for example, 1 arcminute resolution), which indicates that the roll-off is related to the resolution of the initial training samples.

A second correction we apply is to adjust the CIB and radio maps as follows. We re-scale the CIB map by a factor of 1.1 to match the S10 power spectra for $\ell \in (4000, 8000)$. We also re-scale the network CIB and radio source fluxes to match the rescaled S10 source counts.[14] After this re-scaling, we still find that the network overestimates the number counts of CIB sources with flux values greater than 1 mJy; thus we reduce the flux of those sources by replacing the original flux value, S, with $S^{0.63}$. For radio sources, we find that replacing the original flux by $S^{(\lambda)} = ((S+1)^\lambda - 1)/\lambda$ with $\lambda = 1.25$ [108] improves the match at the high-flux end.

E. Generating lensed CMB and multifrequency maps

In order to generate lensed CMB (T, Q, U) maps, we first generate full sky unlensed CMB (T, Q, U) maps as Gaussian random fields on a CAR full-sky grid using the CAMB theory spectra used in S10.[15] Then, these unlensed CMB maps are lensed with a network full-sky κ map using the lensing method described in [4]. As in [4], the lensing operation is performed at 1 arcminute resolution and agrees with the theory to better than few percent up to $\ell = 10{,}000$.

Up to this point, all the training and corrections are done with the 148 GHz network simulations. Here, we discuss how we simulate the other frequency maps. We assume to leading order that the lensed CMB and kSZ perfectly follow a black-body spectrum (i.e., they do not vary with frequency in units of differential CMB temperature). The κ map is frequency independent. In order to simulate the tSZ map at different frequencies, we use the analytic calculations in [109,110] to compute the frequency dependent scaling factor, $a(\nu)$, and scale the 148 GHz tSZ map following $M^{\rm tSZ}(\nu) = a(\nu)/a(148) * M^{\rm tSZ}(148\ {\rm GHz})$, where $M^{\rm tSZ}$ is the real-space tSZ map. For CIB and radio sources, we find the mean spectral indices between each frequency and the reference frequency of 148 GHz, directly from the S10 catalogues; this results in the spectral indices shown in Table III of Appendix C. For each source in the 148 GHz map, we then randomly sample a spectral index from a Gaussian distribution with the mean and the standard deviation given in Table III, and scale its flux accordingly. In this way, we generate full-sky maps of the combined lensed CMB, kSZ, tSZ, CIB, and radio signals at 30, 90, 148, 219, 277, and 350 GHz.

IV. RESULTS

In this section, we study the statistical properties of the mmDL simulations generated by our network. Unless otherwise noted, all statistics are computed using full-sky maps after applying a flux-cut of 7 mJy at 148 GHz to radio and CIB maps, and appling the corrections discussed in Sec. III D. We first visually compare the full-sky mmDL and S10 simulations in Fig. 5. Here we show the full-sky maps in the Mollweide projection, as well as zoom-ins of $1^\circ \times 1^\circ$ patches. Visually, the mmDL simulations and the S10 simulations are indistinguishable.

A. Pixel-level Summary Statistics

There are a number of pixel-level summary statistics used in the literature. Such examples are (1) Minkowski functionals [111], (2) one-point probability distributions, and (3) source counts. For two dimensional images, Minkowski functionals measure the area (F), the contour length (U), and the dimensionless curvature known as the Euler characteristic (χ) as a function of varying pixel threshold value. Combined together these three functionals give an estimate of the topology of an image, which is sensitive to the presence of non-Gaussianity. We compute Minkowski functionals using the *minkfncts2d* software, which is publicly available at [112]. Figure 6 shows as solid curves the functionals computed using the average of 100 $1^\circ \times 1^\circ$ tiles output by the VAEGAN described in Sec. III C.[16] The dashed curves are calculated using the full-sky S10 simulation described in Sec. II. Here, we compute the functionals from the network without the corrections discussed in Sec. III D. Hence, the Minkowski functionals shown in Fig. 6 give an estimate of how well the network performs before applying any corrections. Overall, we find very good agreement between the functionals computed from the network and the S10 simulations. The only noticeable mismatch is in the U and χ functionals of the radio sources, which indicates a mismatch in the real-space shape of radio point sources between the mmDL and S10

[14]Note that the original S10 CIB source fluxes have been rescaled by a factor of 0.75 as discussed in Sec. II A.

[15]The CAMB theory spectra used in this work can be found under the resources directory at [82].

[16]We recomputed the Minkowski functionals on a 20 degree wide band around the celestial equator after making the corrections described in Sec. III D; we did not find any significant change, except a slight shift of the CIB curves as compared to Fig. 6.

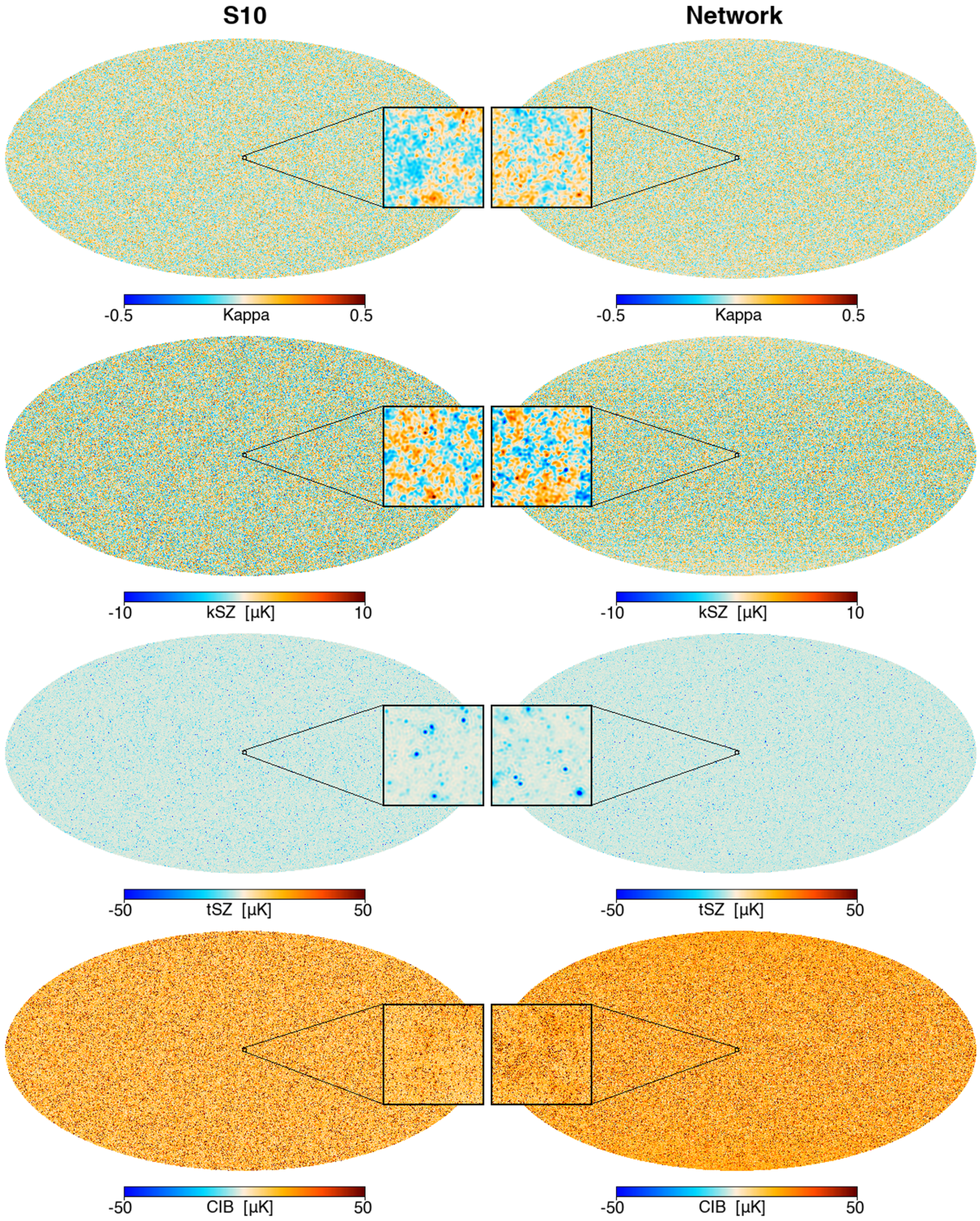

FIG. 5. From top to bottom, shown are the lensing convergence (κ), the kinetic Sunyaev-Zel'dovich effect (kSZ), the thermal Sunyaev-Zel'dovich effect (tSZ), and the cosmic infrared background (CIB) maps at 148 GHz from the S10 simulations (left column) and from the network (right column). A flux cut of 7 mJy at 148 GHz is applied to the CIB maps. Full-sky maps are shown in the Mollweide projection in the background, while center panels show zoom-ins of $1° \times 1°$ patches. All maps have the units of μK, except for the κ map which is dimensionless. Note that the S10 simulations are unique for only one octant of the sky; on the other hand, the network simulations do not have any repeated tiles.

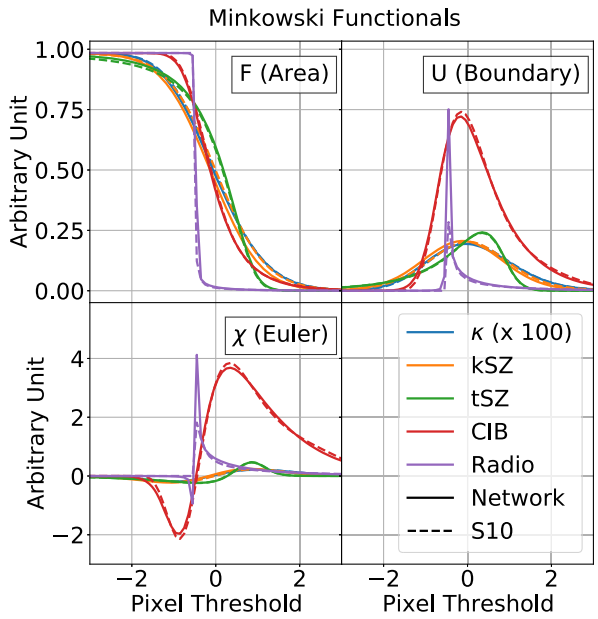

FIG. 6. Shown are the Minkowski functionals of the extragalactic foreground components at 148 GHz. The solid curves are calculated using the average of 100 $1^\circ \times 1^\circ$ tiles output by the VAEGAN described in Sec. III C. The dashed curves are calculated using the full-sky S10 simulation described in Sec. II. Both the S10 and the network simulations are normalized following the procedure detailed in Sec. II C. To compute the Minkowski functionals, a binary map is first constructed with a value of one above a certain pixel threshold, and zero otherwise. Then, the this binary map is used to compute the area above the threshold (F), its boundary (U), and the dimensionless curvature known as the Euler characteristic (χ). These three functionals, give an estimate of the non-Gaussianity of each component [111]. We find that the network can reproduce well the Minkowski functionals calculated using the S10 simulations. This indicates that the network learned the non-Gaussian information in S10.

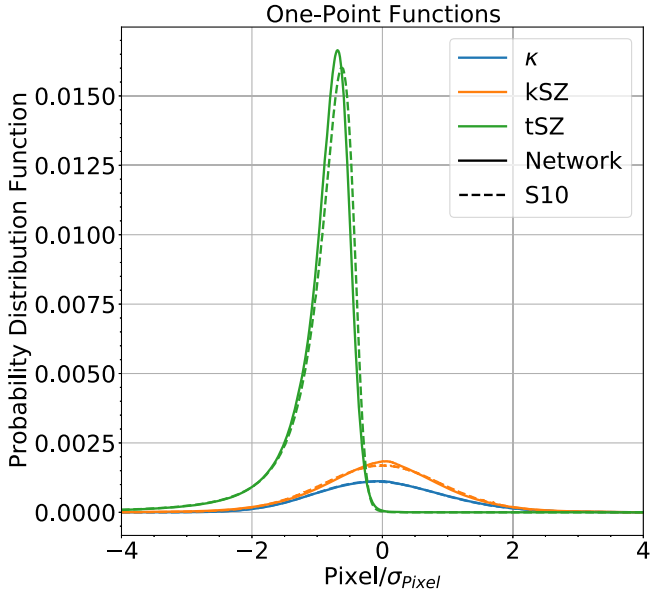

FIG. 7. Shown are one-point probability distribution functions of the extragalactic foregrounds at 148 GHz. Solid curves are calculated from a random sample of a full-sky simulation from the network, and dashed curves are calculated using the full-sky S10 simulation. A flux-cut of 7 mJy is applied to both CIB maps. Each map is divided by the standard deviation of its pixel values in order to have unit variance. We find good agreement between the network curves and the S10 curves.

simulations. The S10 point sources are "true" point sources by construction (i.e., each source falls into a single pixel). On the other hand, the network point sources are constructed as the summation of multiple convolution kernels, hence they can potentially extend beyond a single pixel. In practice, the simulations will be convolved with a beam with a full width at half maximum (FWHM) larger than 0.5 arcminute. So this minor mismatch in the radio source shape will have a negligible effect in actual analyses.

Another pixel-level summary statistic we consider is the one-point probability distribution function (one-point PDF) shown in Fig. 7; this one-point PDF is the histogram of pixel values. Before computing this histogram, we divide the pixels in each component map by the standard deviation of pixel values in that map in order to have unit variance. One-point PDFs have been studied previously in the context of CMB weak lensing and tSZ analyses (e.g., [80,113,114]), and have been shown to be good probes of non-Gaussian information. We find good overall agreement between a random realization of a full-sky network simulation and the S10 simulations.

The last pixel-level statistics we consider is the source number counts. In Fig. 8, we show the histogram of the source number counts per square degree per flux bin for the simulated CIB and radio maps. As discussed in Sec. III D, the network overestimates the number of CIB sources with flux levels greater than 1 mJy, and we correct for this by scaling down the network flux levels at the high flux tail. This scaling substantially improves the overall match. This scaling has negligible impact on the other CIB statistics discussed below, since the other statistics are largely driven by lower flux sources.

B. Two-point statistics

Next, we show the two-point statistics of the simulations in Figs. 9 and 10. Since we apply one-dimensional transfer functions and scale factors, computed by comparing the network power spectra to the S10 power spectra (see Sec. III D for details), the power spectra of the network and S10 match by construction, as shown in Fig. 9. Figure 10 shows the cross spectra between the κ map and the extragalactic foreground components at 148 GHz. Overall, we find a good match between the two sets of cross spectra. We find only a small extra correlation between κ and tSZ at small scales ($\ell \geq 8000$). We also check the cross correlation among the other components (for instance the

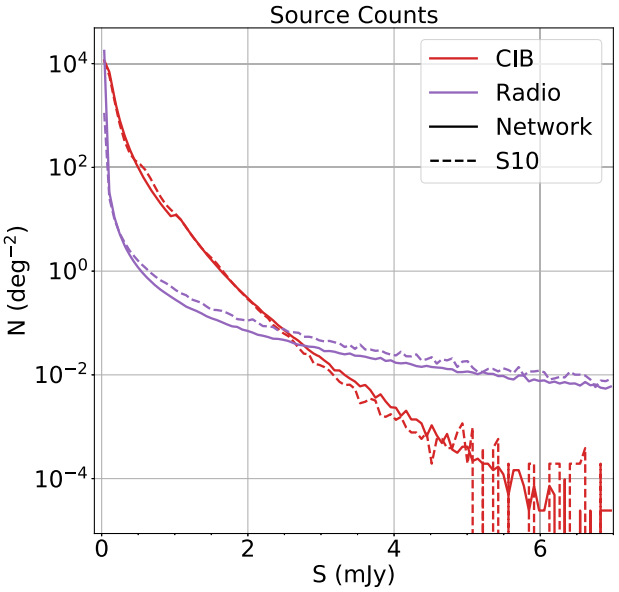

FIG. 8. Shown is a histogram of the source number counts per square degree per flux bin for CIB and radio maps. The solid curves are generated using a random simulation from the network, while the dashed curves are generated using the S10 simulation. A flux cut of 7 mJy is applied to both CIB and radio maps for both network and S10 simulations. We find good overall agreement.

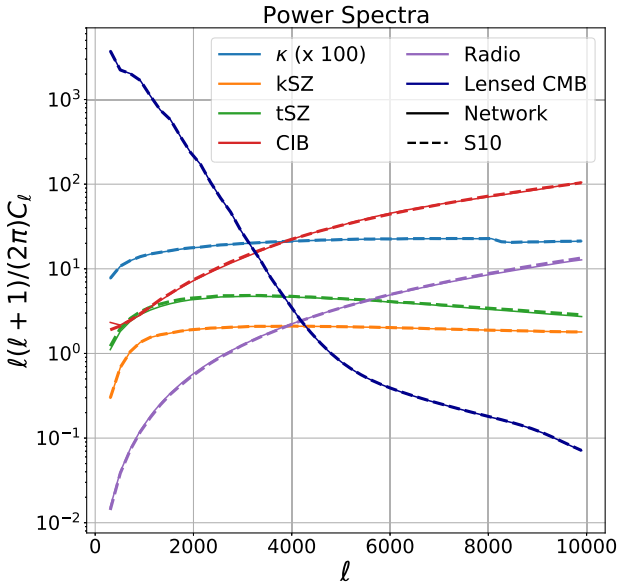

FIG. 9. Shown are the power spectra of simulations at 148 GHz. The solid curves are calculated from a random sample of a full-sky simulation from the network, and dashed curves are calculated using the full-sky S10 simulation. A flux-cut of 7 mJy is applied to both CIB and radio maps. For the purpose of visualization, we multiply the κ map by a factor of 100 before taking the power spectrum. Since we apply one-dimensional transfer functions and scale factors, computed by comparing the network power spectra to the S10 power spectra (see Sec. III D for details), the power spectra of the network and S10 match by construction.

tSZ and CIB), and similarly find good matches. Note that we do not apply any correction to the network cross spectra. The matches between the network and the S10 cross spectra indicate that the network has correctly learned the correlations among the foreground components.

C. Three-point and four-point functions

We also compute higher order statistics, in particular the three-point and four-point functions. We compute the bispectra of foregrounds following [88,115,116] using the *PiInTheSky* software.[17] The binned bispectra of the simulations are shown in Fig. 11 and Fig. 12. One advantage of the binned bispectrum method is that it requires no explicit modeling of the signal. For visualization purposes, we plot the absolute value of the kSZ bispectra, and we plot the tSZ bispectra multiplied by a factor of -1. Overall we find a good match in the shape and amplitude of the bispectra, even though our training procedure was not explicitly optimized to reproduce the bispectra. With minor modifications to the training procedure, we expect this match can be further improved.

In Fig. 12, we show a squeezed configuration bispectra, where one leg of the triangle is $\ell_1 = 35$, and the other two legs have the same length ($\ell_2 = \ell_3$). As discussed in Sec. III D, we generate the full-sky realizations by "stitching" up $1^\circ \times 1^\circ$ tiles. Naively, the network simulations are not expected to reproduce the statistics for modes larger than the tile size (i.e., roughly $\ell \leq 200$). However, we find that the network correctly reproduces the correlations between the super tile modes ($\ell_1 = 35$) and the smaller scale modes ($\ell \geq 200$). This indicates that our overall procedure can generate faithful full-sky realizations, which contain the correct information at all scales.

Figures 13 and 14 shows the comparison for the four-point functions used to reconstruct the lensing potential. To do this comparison, we first generate an unlensed Gaussian CMB realization using the *WMAP5* cosmology [83]; we then make two lensed versions of this CMB map, one lensed by the network κ and another lensed by the S10 κ map. Since our lensing reconstruction filter requires some instrument noise and a finite resolution, we add to each simulation a white noise level of 10 μK-arcminute and a 1.3 arcminute Gaussian beam, which matches current ground-based CMB experiment sensitivity. We reconstruct the lensing potential using the temperature-only quadratic estimator [117] in order to focus on the bias induced from extragalactic foregrounds. We include CMB multipoles in the range of $\ell \in [100, 3000]$ for the reconstruction. Figure 13 shows the cross spectra between the reconstructed lensing convergence and the input lensing convergence from both the network and S10 simulations, in the absence of other foreground contamination.

[17] https://github.com/simonsobs/PiInTheSky.

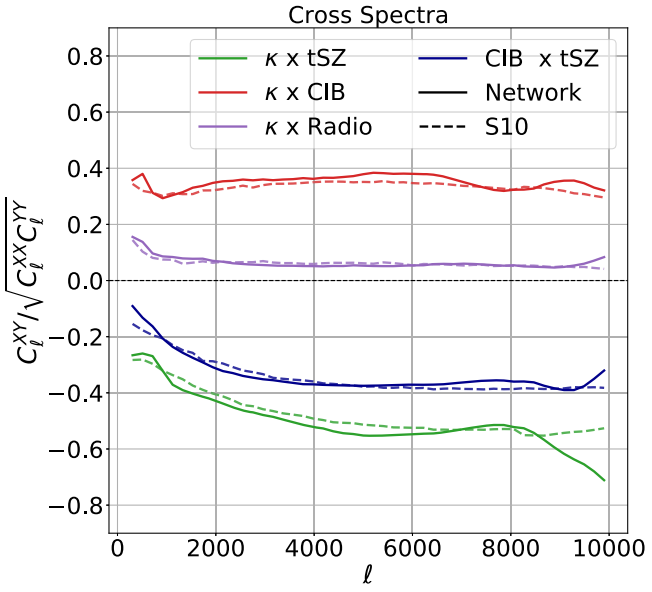

FIG. 10. Shown via cross correlation coefficients are the cross spectra between the κ map and the extragalactic foreground components at 148 GHz. The solid curves are generated with a full-sky map from our network, while the dashed curves are generated from the S10 simulations. A flux-cut of 7 mJy is applied to both the CIB and radio maps. There is a good overall match between the curves. Note that we do not apply any correction to the network cross spectra; the matches between the network and the S10 cross spectra indicate that the network has correctly learned the correlations among the foreground components.

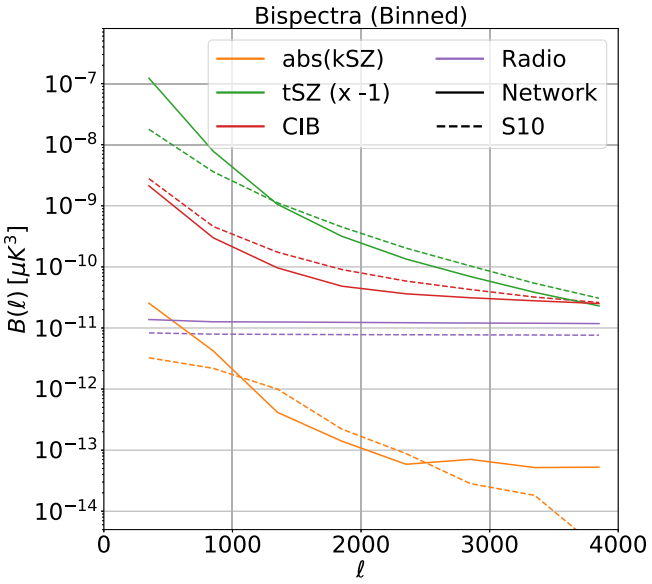

FIG. 11. Shown are the binned bispectra of simulations at 148 GHz. The solid curves are calculated from a random sample of a full-sky simulation from the network, and dashed curves are calculated using the full-sky S10 simulation. A flux-cut of 7 mJy is applied to both CIB and radio maps. Here $\ell = \sqrt{\ell_1^2 + \ell_2^2 + \ell_3^2}$, where ℓ_1, ℓ_2, and ℓ_3 form a triangle.

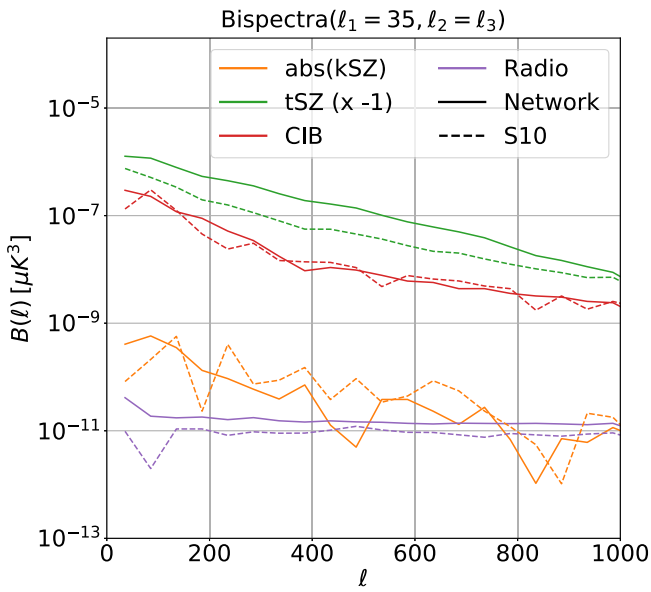

FIG. 12. Similar to Fig. 11, but for a squeezed configuration, where one leg is $\ell_1 = 35$, and the other two legs are the same length ($\ell_2 = \ell_3$). We find that the network can correctly reproduce the correlations between modes larger than the tile size (i.e., $1° \times 1°$ or roughly $\ell \leq 200$ in the harmonic domain) and modes within the tiles (i.e., roughly $\ell \geq 200$).

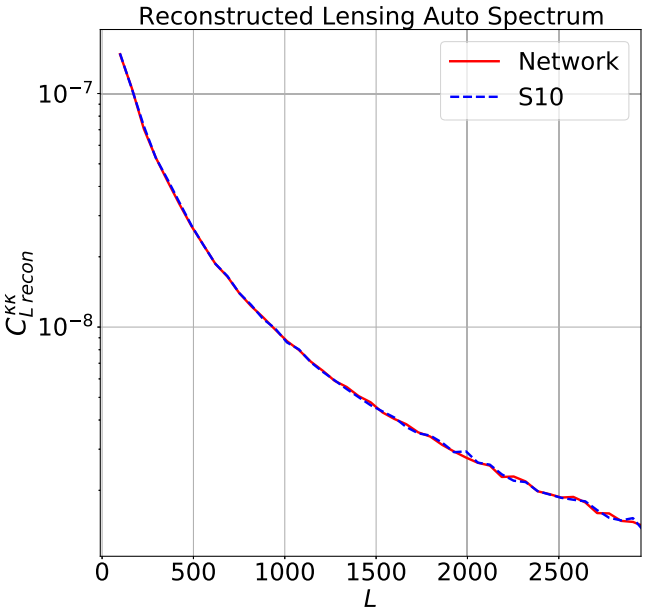

FIG. 13. Shown are the reconstructed lensing auto spectra. The red solid curve is generated from a CMB map lensed with a full-sky κ map from the network. The blue dashed curve is generated using the same CMB map, instead lensed with the full-sky S10 κ. No foregrounds are added. Lensing reconstruction is done using the temperature-only quadratic estimator, including multipoles of $\ell \in [100, 3000]$. We compute the lensing auto spectrum as the cross spectra of the reconstructed kappa map and the input kappa map in order to avoid reconstruction noise bias.

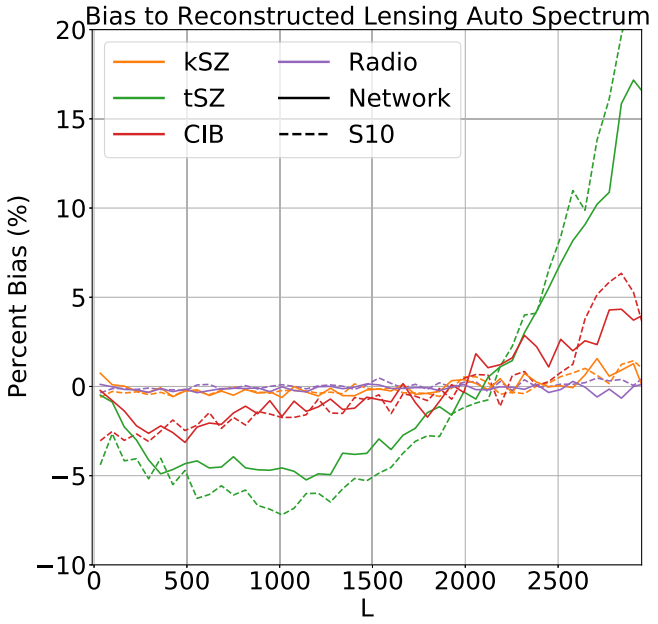

FIG. 14. Shown is the percent bias to the reconstructed lensing auto spectrum induced by extragalactic foregrounds at 148 GHz. The solid curves are calculated with a random full-sky simulation from the network, while the dashed curves are computed using the S10 simulation. Lensed CMB maps are made as discussed in Fig. 13. A flux-cut of 7 mJy is applied to both CIB and radio maps. The lensing auto spectra are computed as the cross spectra of the reconstructed and input kappa maps as in Fig. 13. We find good agreement between the S10 simulation and the network.

We find a good match for all the scales considered. Fig. 14 shows the contamination of the reconstructed lensing convergence from simulated lensed CMB maps plus the extragalactic foregrounds as a percent bias (i.e., $C^{\kappa\kappa}_{L,\mathrm{biased}}/C^{\kappa\kappa}_{L,\mathrm{unbiased}} \times 100$, where $C^{\kappa\kappa}_L$ is computed from the cross spectra between the reconstructed and the input lensing convergences as described above). We find good agreement between the biases computed from the network and those from the S10 simulations.

V. FINAL NETWORK PRODUCT

We make 500 mmDL realizations in total using our network, and release them publicly on the Legacy Archive for Microwave Background Data Analysis (LAMDA)[18] and the National Energy Research Scientific Computing Center (NERSC) cluster.[19] These are full-sky simulations at half-arcminute resolution at six different frequencies (30, 90, 148, 219, 277, and 350 GHz), which include:

(i) the lensing convergence map (κ),
(ii) the kinetic Sunyaev-Zel'dovich effect (kSZ),
(iii) the thermal Sunyaev-Zel'dovich effect (tSZ),
(iv) the cosmic infrared background (CIB), and
(v) the radio galaxies (radio).

For each realization, we generate six maps of the frequency-dependent components (tSZ, CIB, and radio), and one map of the frequency-independent components (lensed CMB (T, Q, U), κ, and kSZ). We release all of these components separately, as well as a combined final microwave sky map in temperature at each of the six frequencies. In total, each realization contains 29 ($6 \times 3 + 5 + 6$) full sky maps. The simulations are natively made in the CAR pixelization and converted to HEALPix by reprojecting the CAR maps to spherical harmonic a_{lm} up to an ℓ_{max} of 10,000 for κ, kSZ, and tSZ maps, and an ℓ_{max} of 25,000 for CIB and radio maps; note that we chose a higher ℓ_{max} of 25,000 for CIB and radio maps to reduce the effect of spectral leakage around the sources and to reproduce the source number counts shown in Fig. 8. These a_{lm} are then reprojected onto full-sky HEALPix maps of Nside = 8192. We release this set of simulations in both HEALPix and CAR pixel formats, however, for space reasons we provide 500 HEALPix realizations and only 30 CAR realizations. We also provide scripts to convert between HEALPix and CAR maps. Each HEALPix fits file is 3.2 GB for double-precision, and each CAR file is 3.6 GB for single-precision. Thus the full package is 50 TB, for $500 \times 29 \times 3.2\,\mathrm{GB} + 30 \times 29 \times 3.6\,\mathrm{GB}$. We also release the code used in this work at [82].

VI. DISCUSSION

We have presented for the first time high-resolution full-sky DL simulations at millimeter wavelengths that include correlated foreground components. We find that these mmDL simulations can reproduce not only the power spectra of the original simulations, but also can naturally reproduce the other non-Gaussian statistics considered in this work (i.e., one-point PDFs, cross-spectra, bispectra, lensing reconstructions, and lensing reconstruction biases) with good overall agreement. This is notable since we only amend the mmDL simulations to reproduce the power spectra and the source number counts above 1 mJy.

The procedure we developed can be used to generate an unlimited number of full-sky realizations starting from just a single realization of a full-sky simulation. Ordinarily there would not be enough independent small tiles from a single full-sky simulation to train a network to predict new foreground maps given a kappa map. However, in this work we achieve this by first generating an augmented dataset (intermediate product 2 in Fig. 1) by training our network on the original simulations. This augmented dataset is more than four times larger than the original one and each patch is statistically independent; thus it can be used to train the next network step (training step 2 in Fig. 1) that requires a larger amount of training data. This enables the capability to mass produce independent full-sky realizations from a single expensive full-sky simulation.

[18] https://lambda.gsfc.nasa.gov/simulation/tb_sim_ov.cfm.
[19] https://crd.lbl.gov/departments/computational-science/c3/c3-research/cosmic-microwave-background/cmb-data-analysis-at-nersc/.

Currently, DL training is mostly done with small tiles due to limited computing resources (both memory and GPU). Though these resource issues will be alleviated as hardware improves overtime, we expect that they will persist for the foreseeable future. Unfortunately, these small DL simulations have a limited use when analyzing current and future wide-field CMB survey data. In this work, we outline a procedure for generating many realizations of full-sky maps, stitching together many individual tiles, that can faithfully recover the high-order statistics of a full-sky map without discontinuities or repeated features.

An advantageous feature of our network is that the input is a full-sky kappa map (top green box in Fig. 1). Thus one can now in principle take a full-sky kappa map from any large-scale structure (LSS) simulation and generate the corresponding lensed CMB and correlated foreground components at millimeter wavelengths. This is especially useful in the current era of combining results from both CMB and LSS surveys, which require a common set of simulations. One can, for example, input in a full-sky kappa map taken from a simulation developed for the Rubin or Roman surveys [118,119], and generate the corresponding CMB maps, resulting in simulations with both galaxies and CMB secondary components.[20]

We envision these mmDL simulations useful for all parts of CMB data analyses, such as verification of pipelines, generation of covariance matrices, and analyses of foreground biases. Furthermore, since our procedure is numerically efficient, especially for generating foreground realizations on a small patch of the sky, we can use our DL method as a part of a forward modeling pipeline (such as discussed in [120]) where one varies the initial conditions to match observations.

Before we conclude, we briefly comment on the potential origin of the corrections we make in Sec. III D. The properties of generative DL models, in particularly those of GAN, are active areas of research (see [121] for more discussion), which is out of the scope of this work. However, we discuss a couple considerations relevant to the applications of generative DL models in physics. First, there is no simple way to impose physical constraints (for instance, symmetry requirements) to neural networks, though there are studies in this direction (for example [122,123]). This prevents DL practitioners from leveraging some physical insights important to the traditional modeling of physical processes. Second, even though the Wasserstein distance used throughout this work is the most general way to compute the distance between two distributions (i.e., the most general discriminator), we can only compute an approximated form of the Wasserstein metric due to numerical reasons. It is not intuitively clear whether optimizing this approximated version should always lead to physical results. These limitations are unlikely fundamental short comings of generative DL models, and we expect that they will be eventually overcome.

We also acknowledge that additional external independent datasets, similar to S10, would be useful for the further validation of the method presented in this work, in particular to assess both the statistical and systematic error inherent to the method. This would also potentially allow us to tune network parameters to match high-order statistics with greater precision. Given the expense of generating such simulations via traditional methods, it is not clear when many such datasets will become available.

Finally, we note that our network is trained to the specific cosmology and baryonic model used to generate the S10 simulations. In principle, a GAN network should be able to generate simulations at different cosmologies and with different gas physics by extrapolating the latent space variables (see [79] for discussion). This is potentially a very powerful technique that can vastly expand the applicability of these mmDL simulations.

[20]If the given LSS kappa map includes non-Gaussian structure, we would just remove training step 3 from our network.

ACKNOWLEDGMENTS

D. H. would like to thank the Center for Computational Astrophysics (CCA) at the Flatiron Institute for support as a CCA pre-doctoral fellow. N. S. would like to thank the CCA for support during her sabbatical, during which time the idea for this project originated. D. H. and N. S. thank Julian Borrill and Andrea Zonca for their help with the product release on NERSC, and Graeme Addison for the product release on NASA LAMBDA. D. H. and N. S. acknowledge support from NSF Grants No. AST-1513618 and No. AST-1907657, and DOE Grant No. DE-SC0020441. F. V. N. acknowledges funding from the Wide Field Infrared Survey Telescope (WFIRST) program through NNG26PJ30C and NNN12AA01C. Some of the results in this paper have been derived using the HEALPix package [89].

APPENDIX A: NORMALIZATION PARAMETERS

In the DL literature, normalization refers to a set of techniques to modify the training data in order to stabilize

TABLE II. The normalization parameters described in Sec. II C. Here σ_p is the standard deviation of pixel values used in the log-normalization [Eq. (1)]. $\mu_{p'}$ and $\sigma_{p'}$ are related to the mean and the standard deviation of the pixel values used in the standard score [Eq. (2)]. Note that the log-normalization is not applied to κ and kSZ maps as discussed in Sec. II C.

Component	σ_p	$\mu_{p'}$	$\sigma_{p'}$
κ	N/A	5.342e−19	0.081
kSZ	N/A	5.173e−17	2.347
tSZ	3.6809	−0.629	0.286
CIB	17.134	0.605	0.349
Radio	6.392	0.199	0.421

and to speed up the training. A few popular normalization methods are the Box-Cox transformation [124] (which includes a log scaling), clipping, min-max, and the standard score. We tried a number of different normalization methods, and found that the combination of log scaling (adopted from [74]), and the standard score yielded the best results in our case. In Table II, we list the parameter values used in our normalization procedure. These parameter values are derived from the validation dataset described in Sec. II B. Each of these parameters are described in detail in Sec. II C.

APPENDIX B: GRID SCHEMATIC

Since our network can only process relatively small tiles, we need a systemic way to divide a full-sky map into a grid of tiles. For numerical reasons, the natural choice of tile width and height is 2^n pixels. In this work, we use the tile size of 128×128 pixels, which is roughly $1° \times 1°$. The schematic of our tile grid is shown in Fig. 15. Note that we include 20 pixels overlap between tiles, which ensures continuity when we reproject tiles on to a full-sky map. This grid procedure results in a grid of tiles with 200 tiles along a longitudinal direction and 400 tiles along a latitude direction.

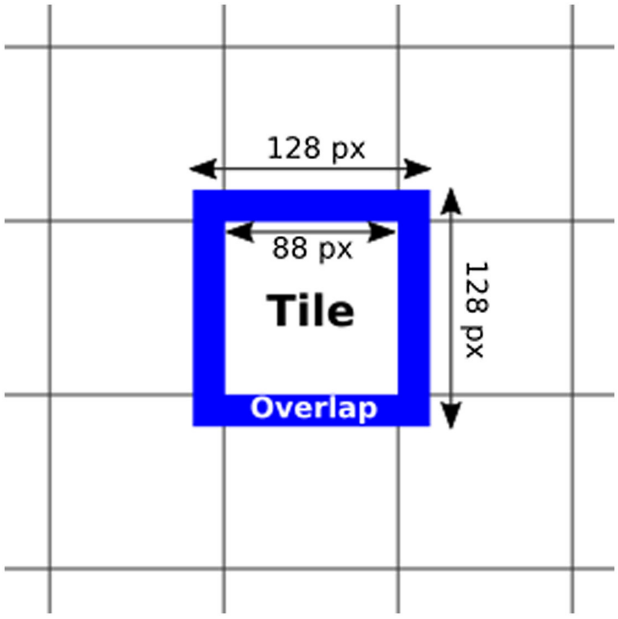

FIG. 15. We show a schematic of the tiling grid described in Sec. III D. Each tile is 128×128 pixels (roughly $1° \times 1°$) in the CAR pixellization and corresponds to a patch effectively centered at the celestial equator. We cutout each tile with a 20-pixel overlap region around the edges to ensure that the final maps are continuous when we reproject the tiles back on to a full-sky CAR map.

TABLE III. The estimated spectral indices of CIB and radio sources described in Sec. III D and Appendix C. The spectral indices are estimated using the S10 source catalogues by comparing the flux of a source at a given frequency (30, 90, 219, 277, and 350 GHz) to the flux at the reference frequency of 148 GHz; the mean and standard deviation of the source spectral indices are listed in the Table.

Frequency (GHz)	$\beta_{\rm CIB}$	$\beta_{\rm radio}$
30	3.304 ± 0.138	-0.799 ± 0.158
90	3.202 ± 0.442	-0.811 ± 0.144
219	2.973 ± 0.567	-0.820 ± 0.135
277	2.870 ± 0.372	-0.822 ± 0.134
350	2.745 ± 0.301	-0.823 ± 0.133

APPENDIX C: FREQUENCY DEPENDENCE OF CIB AND RADIO SOURCES

As discussed in Sec. III D, we estimate the frequency dependence of CIB and radio sources using the S10 catalogues. For each source in the catalogues, we compute its spectral index between a given frequency and the reference frequency of 148 GHz by comparing the flux of that source at the given frequency (30, 90, 219, 277, and 350 GHz) to the flux at the reference frequency. In Table III, we show the means and standard deviations

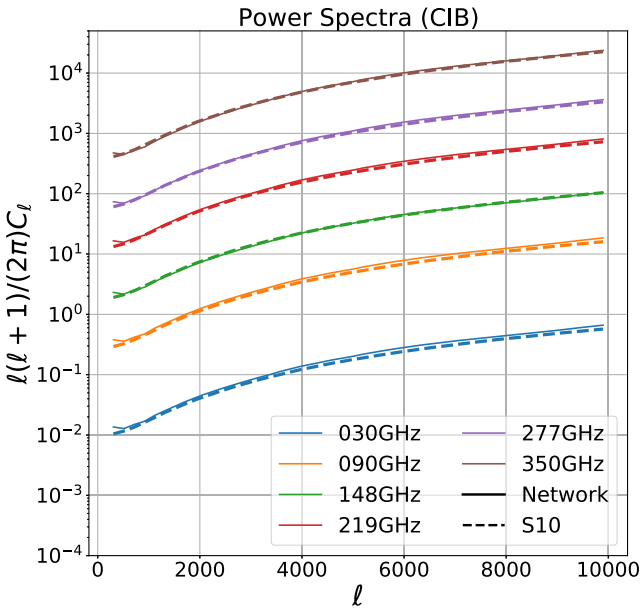

FIG. 16. We show the power spectra of CIB maps at different frequencies. The solid curves are calculated from a random realization of a full-sky simulation from the network, and dashed curves are calculated using the full-sky S10 simulation. A flux-cut of 7 mJy at 148 GHz is applied to all CIB maps. For each source in the 148 GHz map, we simulate its frequency dependence as described in Sec. III D and Appendix C. We find good agreement for the CIB power spectra between the S10 simulation and the network.

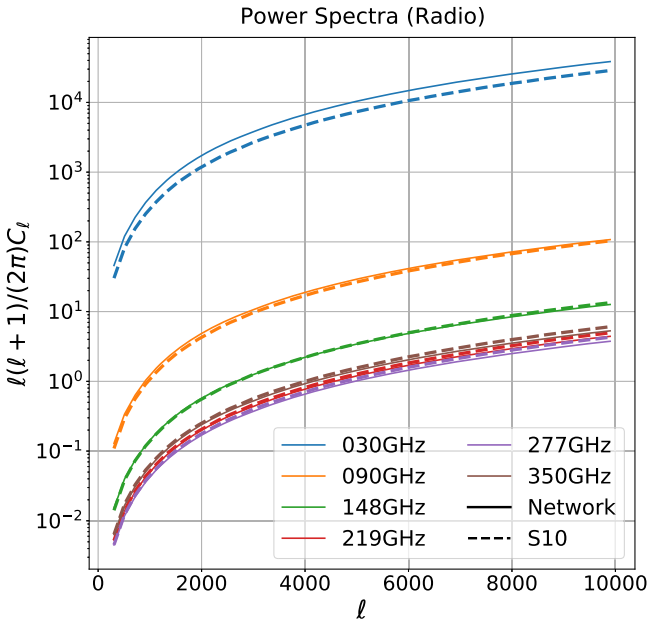

FIG. 17. The same as Fig. 16 above, but for radio maps instead of CIB maps.

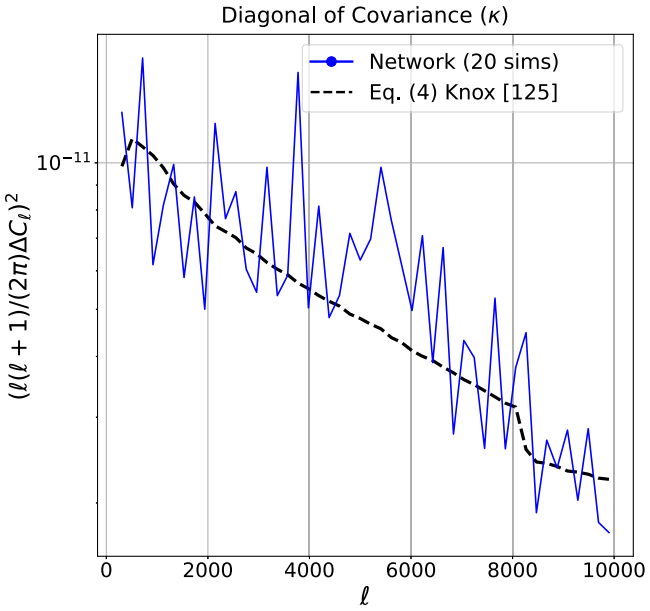

FIG. 18. We show the diagonal elements of the covariance matrix derived from the κ power spectrum. The blue curves are estimated from 20 realizations of a full-sky simulation from the network, and the dashed curve is calculated using Eq. (4) in [125] given the S10 κ power spectrum. We find good agreement between the variance from the network simulations and the expected variance.

of the spectral indices for CIB and radio sources. Note that we only consider the sources that meet our flux-cut criteria ($\leq$7 mJy at 148 GHz) in this estimation.

For each source in the network 148 GHz map, we randomly assign it a spectral index for each frequency, with each index drawn from a Gaussian distribution with a mean and the standard deviation as specified in Table III; we then scale the source's 148 GHz flux accordingly and place it in the new frequency map. We find that we can recover well the frequency dependence of the original S10 CIB and radio maps using this method as shown in Figs. 16 and 17.

APPENDIX D: REALIZATION DEPENDENT SCATTER

One common failure mode of GANs is mode collapse, which is when a GAN repeatedly generates the same or statistically similar images. In the context of our work, mode collapse will lead to a lower variance of the simulations than expected, due to the presence of repeated tiles. In this section, we empirically show that the variances of our network simulations match the expected variances. Figure 18 shows the diagonal elements of the covariance matrix derived from the κ power spectrum of 20 network simulations. Here, the expected variance is given by Eq. (4) in [125]; in the absence of experimental noise, this equation gives the expected variance of an approximately Gaussian random field. Similarly, we show the estimated variance of 20 network kSZ and CIB maps in Figs. 19 and 20. For each of these components, we find that the variance from the network simulations matches the expected variance. Since each lensed CMB realization is generated from independent unlensed CMB realizations, the lensed CMB maps should have the correct variance by the construction.

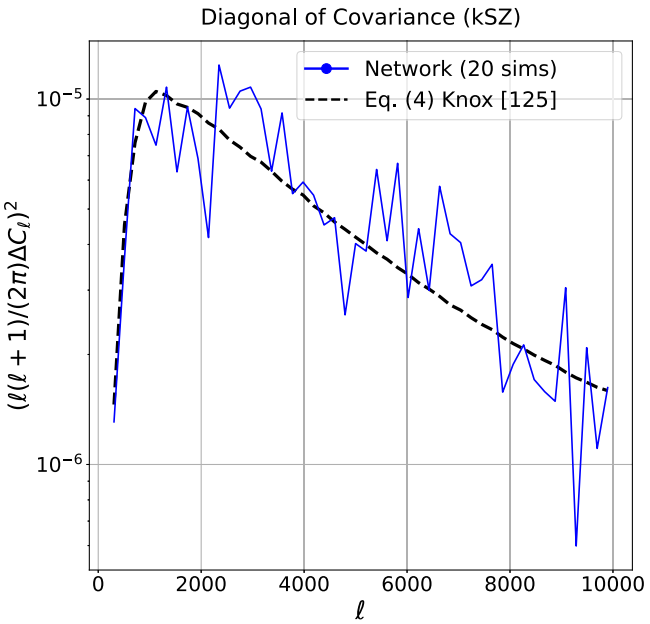

FIG. 19. The same as Fig. 18 above, but for kSZ maps instead of κ maps.

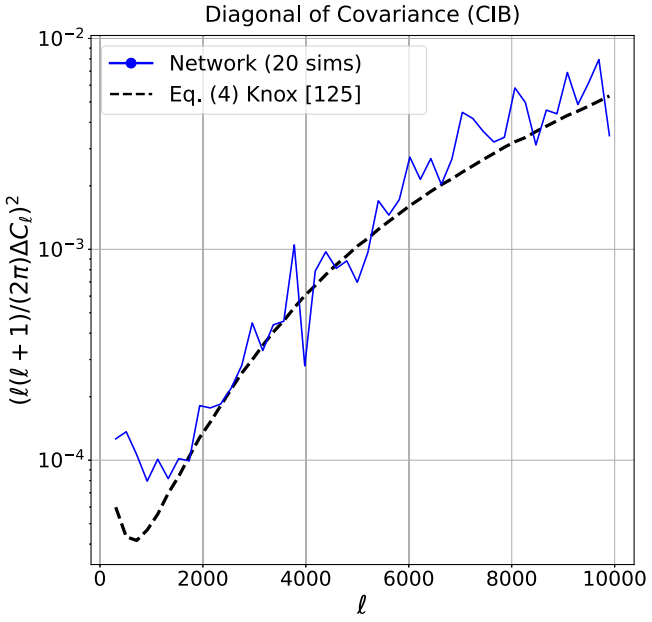

FIG. 20. The same as Fig. 18 above, but for CIB maps at 148 GHz instead of κ maps. Note that a flux-cut of 7 mJy at 148 GHz is applied to the CIB maps.

APPENDIX E: SUPPLEMENTARY NETWORK TRAINING DETAILS

The details of our network architectures are discussed in Sec. III. In this section, we provide supplementary details of our training procedure. Table IV summarizes a few additional parameters of our network training procedure. As shown in the table, all three GAN layers share most of the common parameters including the learning rate and the Adam optimizer parameters. We check our metrics (images, loss function, power spectra, and others) every 100,000 training data samples processed (i.e., two times per epoch for DCWGAN-GP and VAEGAN; eight times per epoch for PIXGAN). If we find a degradation of the output quality, we restart the training from the last epoch with the learning rate reduced by a factor of two. Also, if the model either can not reproduce the power spectra after the loss function plateaus or images are not visually correct, we start the training from scratch with the learning rate reduced by a factor of two. Lastly, we do not explicitly check for mode collapse, but rely on the summary statistics (shown in Sec. IV) and the estimated variances (shown in Appendix D) to infer the presence of mode collapse.

TABLE IV. Shown are additional parameters used to train the three GAN layers discussed in Sec. III.

	DCWGAN-GP	PIXGAN	VAEGAN
Type	Nonconditional	Conditional	Conditional
Loss function	Wasserstein		
Optimize	Adam optimizer ($\beta_1 = 0.5$, $\beta_2 = 0.9$)		
Learning rate	Initially $lr = 10^{-4}$. Halved if needed		
Batch size	32		

[1] N. Aghanim, Y. Akrami, M. Ashdown, J. Aumont, C. Baccigalupi, M. Ballardini, A. J. Banday, R. B. Barreiro, N. Bartolo *et al.* (Planck Collaboration), Astron. Astrophys. **641**, A6 (2020).

[2] J. W. Henning, J. T. Sayre, C. L. Reichardt, P. A. R. Ade, A. J. Anderson, J. E. Austermann, J. A. Beall, A. N. Bender, B. A. Benson, L. E. Bleem *et al.*, Astrophys. J. **852**, 97 (2018).

[3] S. Aiola, E. Calabrese, L. Maurin, S. Naess, B. L. Schmitt, M. H. Abitbol, G. E. Addison, P. A. R. Ade, D. Alonso, M. Amiri *et al.*, J. Cosmol. Astropart. Phys. 12 (2020) 047.

[4] S. K. Choi, M. Hasselfield, S.-P. P. Ho, B. Koopman, M. Lungu, M. H. Abitbol, G. E. Addison, P. A. R. Ade, S. Aiola, D. Alonso *et al.*, J. Cosmol. Astropart. Phys. 12 (2020) 045.

[5] D. Han, N. Sehgal, A. MacInnis, A. van Engelen, B. D. Sherwin, M. S. Madhavacheril, S. Aiola, N. Battaglia, J. A. Beall, D. T. Becker *et al.*, J. Cosmol. Astropart. Phys. 01 (2021) 031.

[6] B. D. Sherwin, A. van Engelen, N. Sehgal, M. Madhavacheril, G. E. Addison, S. Aiola, R. Allison, N. Battaglia, D. T. Becker, J. A. Beall *et al.*, Phys. Rev. D **95**, 123529 (2017).

[7] N. Aghanim, Y. Akrami, M. Ashdown, J. Aumont, C. Baccigalupi, M. Ballardini, A. J. Banday, R. B. Barreiro, N. Bartolo *et al.* (Planck Collaboration), Astron. Astrophys. **641**, A8 (2020).

[8] W. L. K. Wu, L. M. Mocanu, P. A. R. Ade, A. J. Anderson, J. E. Austermann, J. S. Avva, J. A. Beall, A. N. Bender, B. A. Benson, F. Bianchini *et al.*, Astrophys. J. **884**, 70 (2019).

[9] P. A. R. Ade, N. Aghanim, M. Arnaud, M. Ashdown, J. Aumont, C. Baccigalupi, A. J. Banday, R. B. Barreiro, J. G. Bartlett *et al.* (Planck Collaboration), Astron. Astrophys. **594**, A24 (2016).

[10] S. Amodeo, N. Battaglia, E. Schaan, S. Ferraro, E. Moser, S. Aiola, J. E. Austermann, J. A. Beall, R. Bean, D. T. Becker *et al.*, Phys. Rev. D **103**, 063514 (2021).

[11] E. Schaan, S. Ferraro, S. Amodeo, N. Battaglia, S. Aiola, J. E. Austermann, J. A. Beall, R. Bean, D. T. Becker, R. J. Bond *et al.*, Phys. Rev. D **103**, 063513 (2021).

[12] N. Huang, L. E. Bleem, B. Stalder, P. A. R. Ade, S. W. Allen, A. J. Anderson, J. E. Austermann, J. S. Avva, J. A. Beall, A. N. Bender *et al.*, Astron. J. **159**, 110 (2020).

[13] A. H. Jaffe, E. Gawiser, D. Finkbeiner, J. C. Baker, A. Balbi, M. Davis, S. Hanany, W. Holzapfel, M. Krumholz, L. Moustakas *et al.*, in *Microwave Foregrounds*, edited by A. de Oliveira-Costa and M. Tegmark, Astronomical Society of the Pacific Conference Series, Vol. 181 (Astronomical Society of the Pacific, San Francisco, 1999), p. 367.
[14] J. González-Nuevo, L. Toffolatti, and F. Argüeso, Astrophys. J. **621**, 1 (2005).
[15] B. M. Schäfer, C. Pfrommer, M. Bartelmann, V. Springel, and L. Hernquist, Mon. Not. R. Astron. Soc. **370**, 1309 (2006).
[16] M. Roncarelli, L. Moscardini, S. Borgani, and K. Dolag, Mon. Not. R. Astron. Soc. **378**, 1259 (2007).
[17] C. Carbone, V. Springel, C. Baccigalupi, M. Bartelmann, and S. Matarrese, Mon. Not. R. Astron. Soc. **388**, 1618 (2008).
[18] N. Battaglia, J. R. Bond, C. Pfrommer, J. L. Sievers, and D. Sijacki, Astrophys. J. **725**, 91 (2010).
[19] N. Sehgal, P. Bode, S. Das, C. Hernand ez-Monteagudo, K. Huffenberger, Y.-T. Lin, J. P. Ostriker, and H. Trac, Astrophys. J. **709**, 920 (2010).
[20] S. Flender, L. Bleem, H. Finkel, S. Habib, K. Heitmann, and G. Holder, Astrophys. J. **823**, 98 (2016).
[21] R. Takahashi, T. Hamana, M. Shirasaki, T. Namikawa, T. Nishimichi, K. Osato, and K. Shiroyama, Astrophys. J. **850**, 24 (2017).
[22] G. Stein, M. A. Alvarez, J. R. Bond, A. van Engelen, and N. Battaglia, J. Cosmol. Astropart. Phys. 10 (2020) 012.
[23] J. W. Fowler, M. D. Niemack, S. R. Dicker, A. M. Aboobaker, P. A. R. Ade, E. S. Battistelli, M. J. Devlin, R. P. Fisher, M. Halpern, P. C. Hargrave *et al.*, Appl. Opt. **46**, 3444 (2007).
[24] R. J. Thornton, P. A. R. Ade, S. Aiola, F. E. Angilè, M. Amiri, J. A. Beall, D. T. Becker, H. M. Cho, S. K. Choi, P. Corlies *et al.*, Astrophys. J. Suppl. Ser. **227**, 21 (2016).
[25] J. E. Carlstrom, P. A. R. Ade, K. A. Aird, B. A. Benson, L. E. Bleem, S. Busetti, C. L. Chang, E. Chauvin, H. M. Cho, T. M. Crawford *et al.*, Publ. Astron. Soc. Pac. **123**, 568 (2011).
[26] P. Ade, J. Aguirre, Z. Ahmed, S. Aiola, A. Ali, D. Alonso, M. A. Alvarez, K. Arnold, P. Ashton, J. Austermann *et al.*, J. Cosmol. Astropart. Phys. (2019) 056.
[27] K. Abazajian, G. Addison, P. Adshead, Z. Ahmed, S. W. Allen, D. Alonso, M. Alvarez, A. Anderson, K. S. Arnold, C. Baccigalupi *et al.*, arXiv:1907.04473.
[28] N. Sehgal, S. Aiola, Y. Akrami, K. Basu, M. Boylan-Kolchin, S. Bryan, S. Clesse, F.-Y. Cyr-Racine, L. Di Mascolo, S. Dicker *et al.*, Bull. Am. Astron. Soc. **51**, 6 (2019).
[29] N. Sehgal, S. Aiola, Y. Akrami, K. moni Basu, Boylan-Kolchin, S. Bryan, C. M. Casey, S. Clesse, F.-Y. Cyr-Racine, L. Di Mascolo *et al.*, arXiv:2002.12714.
[30] A. Lewis, Phys. Rev. D **71**, 083008 (2005).
[31] A. Amblard, C. Vale, and M. White, New Astron. **9**, 687 (2004).
[32] S. Basak, S. Prunet, and K. Benabed, arXiv:0811.1677.
[33] G. Fabbian and R. Stompor, Astron. Astrophys. **556**, A109 (2013).
[34] S. K. Næss and T. Louis, J. Cosmol. Astropart. Phys. 09 (2013) 001.
[35] C. Baccigalupi, New Astron. Rev. **47**, 1127 (2003).
[36] P. Hennebelle, R. Banerjee, E. Vázquez-Semadeni, R. S. Klessen, and E. Audit, Astron. Astrophys. **486**, L43 (2008).
[37] A. Waelkens, T. Jaffe, M. Reinecke, F. S. Kitaura, and T. A. Enßlin, Astron. Astrophys. **495**, 697 (2009).
[38] S. K. Choi and L. A. Page, J. Cosmol. Astropart. Phys. 12 (2015) 020.
[39] B. Thorne, J. Dunkley, D. Alonso, and S. Næss, Mon. Not. R. Astron. Soc. **469**, 2821 (2017).
[40] F. Vansyngel, F. Boulanger, T. Ghosh, B. Wandelt, J. Aumont, A. Bracco, F. Levrier, P. G. Martin, and L. Montier, Astron. Astrophys. **603**, A62 (2017).
[41] D. Beck, J. Errard, and R. Stompor, J. Cosmol. Astropart. Phys. 06 (2020) 030.
[42] K. Aylor, M. Haq, L. Knox, Y. Hezaveh, and L. Perreault-Levasseur, Mon. Not. R. Astron. Soc. **500**, 3889 (2020).
[43] B. Thorne, L. Knox, and K. Prabhu, Mon. Not. R. Astron. Soc. **504**, 2603 (2021).
[44] N. Krachmalnicoff and G. Puglisi, Astrophys. J. **911**, 42 (2021).
[45] L. Perreault Levasseur, Y. D. Hezaveh, and R. H. Wechsler, Astrophys. J. Lett. **850**, L7 (2017).
[46] S. Ravanbakhsh, J. Oliva, S. Fromenteau, L. C. Price, S. Ho, J. Schneider, and B. Poczos, arXiv:1711.02033.
[47] A. Gupta, J. M. Zorrilla Matilla, D. Hsu, and Z. Haiman, Phys. Rev. D **97**, 103515 (2018).
[48] A. Mathuriya, D. Bard, P. Mendygral, L. Meadows, J. Arnemann, L. Shao, S. He, T. Karna, D. Moise, S. J. Pennycook *et al.*, arXiv:1808.04728.
[49] J. Alsing, T. Charnock, S. Feeney, and B. Wandelt, Mon. Not. R. Astron. Soc. **488**, 5093 (2019).
[50] M. Shirasaki, K. Moriwaki, T. Oogi, N. Yoshida, S. Ikeda, and T. Nishimichi, Mon. Not. R. Astron. Soc. **504**, 1825 (2021).
[51] L. Lucie-Smith, H. V. Peiris, A. Pontzen, B. Nord, and J. Thiyagalingam, arXiv:2011.10577.
[52] F. Villaescusa-Navarro, B. D. Wandelt, D. Anglés-Alcázar, S. Genel, J. M. Zorrilla Mantilla, S. Ho, and D. N. Spergel, arXiv:2011.05992.
[53] M. A. Petroff, G. E. Addison, C. L. Bennett, and J. L. Weiland, Astrophys. J. **903**, 104 (2020).
[54] K. Moriwaki, M. Shirasaki, and N. Yoshida, Astrophys. J. Lett. **906**, L1 (2021).
[55] J. Caldeira, W. L. K. Wu, B. Nord, C. Avestruz, S. Trivedi, and K. T. Story, Astron. Comput. **28**, 100307 (2019).
[56] V. Bonjean, Astron. Astrophys. **634**, A81 (2020).
[57] E. Guzman and J. Meyers, Phys. Rev. D **104**, 043529 (2021).
[58] Z. Lin, N. Huang, C. Avestruz, W. L. K. Wu, S. Trivedi, J. Caldeira, and B. Nord, Mon. Not. R. Astron. Soc. **507**, 4149 (2021).
[59] G. Puglisi and X. Bai, Astrophys. J. **905**, 143 (2020).
[60] G. Montefalcone, M. H. Abitbol, D. Kodwani, and R. D. P. Grumitt, J. Cosmol. Astropart. Phys. 03 (2021) 055.
[61] A. Vafaei Sadr and F. Farsian, J. Cosmol. Astropart. Phys. 03 (2021) 012.
[62] M. Münchmeyer and K. M. Smith, arXiv:1905.05846.
[63] L. Lucie-Smith, H. V. Peiris, A. Pontzen, and M. Lochner, Mon. Not. R. Astron. Soc. **479**, 3405 (2018).

[64] C. C. Lovell, V. Acquaviva, P. A. Thomas, K. G. Iyer, E. Gawiser, and S. M. Wilkins, Mon. Not. R. Astron. Soc. **490**, 5503 (2019).
[65] R. Alves de Oliveira, Y. Li, F. Villaescusa-Navarro, S. Ho, and D. N. Spergel, arXiv:2012.00240.
[66] B. H. Chen, T. Goto, S. J. Kim, T. W. Wang, D. J. D. Santos, S. C.-C. Ho, T. Hashimoto, A. Poliszczuk, A. Pollo, S. Trippe *et al.*, Mon. Not. R. Astron. Soc. **501**, 3951 (2021).
[67] M. Cranmer, A. Sanchez-Gonzalez, P. Battaglia, R. Xu, K. Cranmer, D. Spergel, and S. Ho, arXiv:2006.11287.
[68] D. Wadekar, F. Villaescusa-Navarro, S. Ho, and L. Perreault-Levasseur, arXiv:2012.00111.
[69] G.-J. Wang, X.-J. Ma, and J.-Q. Xia, Mon. Not. R. Astron. Soc. **501**, 5714 (2021).
[70] P. Berger and G. Stein, Mon. Not. R. Astron. Soc. **482**, 2861 (2019).
[71] S. He, Y. Li, Y. Feng, S. Ho, S. Ravanbakhsh, W. Chen, and B. Póczos, Proc. Natl. Acad. Sci. U.S.A. **116**, 13825 (2019).
[72] M. Mustafa, D. Bard, W. Bhimji, Z. Lukić, R. Al-Rfou, and J. M. Kratochvil, Comput. Astrophys. Cosmol. **6**, 1 (2019).
[73] N. Perraudin, A. Srivastava, A. Lucchi, T. Kacprzak, T. Hofmann, and A. Réfrégier, Comput. Astrophys. Cosmol. **6**, 5 (2019).
[74] T. Tröster, C. Ferguson, J. Harnois-Déraps, and I. G. McCarthy, Mon. Not. R. Astron. Soc. **487**, L24 (2019).
[75] C. Chen, Y. Li, F. Villaescusa-Navarro, S. Ho, and A. Pullen, arXiv:2012.05472.
[76] O. Curtis and T. G. Brainerd, Res. Notes Am. Astron. Soc. **4**, 90 (2020).
[77] B. Dai and U. Seljak, Proc. Natl. Acad. Sci. U.S.A. **118**, e2020324118 (2021).
[78] Y. Li, Y. Ni, R. A. C. Croft, T. Di Matteo, S. Bird, and Y. Feng, Proc. Natl. Acad. Sci. U.S.A. **118**, e2022038118 (2021).
[79] A. Tamosiunas, H. A. Winther, K. Koyama, D. J. Bacon, R. C. Nichol, and B. Mawdsley, Mon. Not. R. Astron. Soc. **506**, 3049 (2021).
[80] L. Thiele, J. C. Hill, and K. M. Smith, Phys. Rev. D **102**, 123545 (2020).
[81] M. Ullmo, A. Decelle, and N. Aghanim, Astron. Astrophys. **651**, A46 (2021).
[82] https://github.com/dwhan89/cosmikyu.
[83] J. Dunkley, E. Komatsu, M. R. Nolta, D. N. Spergel, D. Larson, G. Hinshaw, L. Page, C. L. Bennett, B. Gold, N. Jarosik *et al.*, Astrophys. J. Suppl. Ser. **180**, 306 (2009).
[84] F. Lacasa, N. Aghanim, M. Kunz, and M. Frommert, Mon. Not. R. Astron. Soc. **421**, 1982 (2012).
[85] F. Lacasa, A. Pénin, and N. Aghanim, Mon. Not. R. Astron. Soc. **439**, 123 (2014).
[86] N. Hand, G. E. Addison, E. Aubourg, N. Battaglia, E. S. Battistelli, D. Bizyaev, J. R. Bond, H. Brewington, J. Brinkmann, B. R. Brown *et al.*, Phys. Rev. Lett. **109**, 041101 (2012).
[87] J. C. Hill and B. D. Sherwin, Phys. Rev. D **87**, 023527 (2013).
[88] A. van Engelen, S. Bhattacharya, N. Sehgal, G. P. Holder, O. Zahn, and D. Nagai, Astrophys. J. **786**, 13 (2014).
[89] K. M. Górski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann, Astrophys. J. **622**, 759 (2005).
[90] J. Dunkley, E. Calabrese, J. Sievers, G. E. Addison, N. Battaglia, E. S. Battistelli, J. R. Bond, S. Das, M. J. Devlin, R. Dünner *et al.*, J. Cosmol. Astropart. Phys. 07 (2013) 025.
[91] J. L. Sievers, R. A. Hlozek, M. R. Nolta, V. Acquaviva, G. E. Addison, P. A. R. Ade, P. Aguirre, M. Amiri, J. W. Appel, L. F. Barrientos *et al.*, J. Cosmol. Astropart. Phys. 10 (2013) 060.
[92] P. A. R. Ade, N. Aghanim, C. Armitage-Caplan, M. Arnaud, M. Ashdown, F. Atrio-Barandela, J. Aumont, C. Baccigalupi, A. J. Banday *et al.* (Planck Collaboration), Astron. Astrophys. **571**, A21 (2014).
[93] E. M. George, C. L. Reichardt, K. A. Aird, B. A. Benson, L. E. Bleem, J. E. Carlstrom, C. L. Chang, H. M. Cho, T. M. Crawford, A. T. Crites *et al.*, Astrophys. J. **799**, 177 (2015).
[94] N. Aghanim, M. Arnaud, M. Ashdown, J. Aumont, C. Baccigalupi, A. J. Banday, R. B. Barreiro, J. G. Bartlett, N. Bartolo *et al.* (Planck Collaboration), Astron. Astrophys. **594**, A22 (2016).
[95] M. Reinecke and D. S. Seljebotn, Astron. Astrophys. **554**, A112 (2013).
[96] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, arXiv:1406.2661.
[97] I. Goodfellow, arXiv:1701.00160.
[98] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, IEEE Signal Process. Mag. **35**, 53 (2018).
[99] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, arXiv:1704.00028.
[100] B. Xu, N. Wang, T. Chen, and M. Li, arXiv:1505.00853.
[101] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, arXiv:1811.03378
[102] D. P. Kingma and J. Ba, arXiv:1412.6980.
[103] O. Ronneberger, P. Fischer, and T. Brox, arXiv:1505.04597.
[104] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, arXiv:1611.07004.
[105] C. Dong, C. C. Loy, K. He, and X. Tang, in *Computer Vision —ECCV 2014*, edited by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars (Springer International Publishing, Cham, 2014), pp. 184–199, ISBN 978-3-319-10593-2.
[106] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, arXiv:1512.09300.
[107] S. Naess, S. Aiola, J. E. Austermann, N. Battaglia, J. A. Beall, D. T. Becker, R. J. Bond, E. Calabrese, S. K. Choi, N. F. Cothard *et al.*, J. Cosmol. Astropart. Phys. 12 (2020) 046.
[108] I. Yeo and R. A. Johnson, Biometrika **87**, 954 (2000).
[109] Y. B. Zeldovich and R. A. Sunyaev, Astrophys. Space Sci. **4**, 301 (1969).
[110] R. A. Sunyaev and Y. B. Zeldovich, Astrophys. Space Sci. **7**, 3 (1970).
[111] H. Mantz, K. Jacobs, and K. Mecke, J. Stat. Mech. (2008), P12015.
[112] https://github.com/moutazhaq/minkfncts2d.

[113] J. C. Hill, B. D. Sherwin, K. M. Smith, G. E. Addison, N. Battaglia, E. S. Battistelli, J. R. Bond, E. Calabrese, M. J. Devlin, J. Dunkley *et al.*, arXiv:1411.8004.
[114] J. Liu, J. C. Hill, B. D. Sherwin, A. Petri, V. Böhm, and Z. Haiman, Phys. Rev. D **94**, 103501 (2016).
[115] T. M. Crawford, K. K. Schaffer, S. Bhattacharya, K. A. Aird, B. A. Benson, L. E. Bleem, J. E. Carlstrom, C. L. Chang, H. M. Cho, A. T. Crites *et al.*, Astrophys. J. **784**, 143 (2014).
[116] W. R. Coulton and D. N. Spergel, J. Cosmol. Astropart. Phys. 10 (2019) 056.
[117] W. Hu and T. Okamoto, Astrophys. J. **574**, 566 (2002).
[118] D. Spergel, N. Gehrels, C. Baltay, D. Bennett, J. Breckinridge, M. Donahue, A. Dressler, B. S. Gaudi, T. Greene, O. Guyon *et al.*, arXiv:1503.03757.
[119] Ž. Ivezić, S. M. Kahn, J. A. Tyson, B. Abel, E. Acosta, R. Allsman, D. Alonso, Y. AlSayyad, S. F. Anderson, J. Andrew *et al.*, Astrophys. J. **873**, 111 (2019).
[120] M. Millea, C. M. Daley, T.-L. Chou, E. Anderes, P. A. R. Ade, A. J. Anderson, J. E. Austermann, J. S. Avva, J. A. Beall, A. N. Bender *et al.*, arXiv:2012.01709.
[121] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, arXiv:1811.10597.
[122] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris, J. Comput. Phys. **394**, 56 (2019).
[123] J.-L. Wu, K. Kashinath, A. Albert, D. Chirila, Prabhat, and H. Xiao, J. Comput. Phys. **406**, 109209 (2020).
[124] G. E. P. Box and D. R. Cox, J. R. Stat. Soc. Ser. B Stat. Methodol. **26**, 211 (1964).
[125] L. Knox, Phys. Rev. D **52**, 4307 (1995).