Teaching a machine to generate an artificial universe

Jed Homer*[1]

[1]Institute for Gravitational Research, School of Physics and Astronomy, University of Glasgow, G12 8QQ, UK

Abstract

This work presents separate two- and three-dimensional generative frameworks that use Generative Adversarial Networks to create realistic samples of cosmological structure. Provisional samples of the virtual cosmic web are obtained from N-body simulation data that is restricted to volumes and projections of size $80\,h^{-1}$ Mpc. The generated samples are shown to demonstrate a high degree of coherence with their simulated counterparts and the distinction cannot be made visually by human experts. The nature of the separation between generated and N-body simulation samples is quantified with repeated tests of the Kullback-Leibler divergence on ensembles of samples alongside samples with uniformly distributed pixel densities. It is found that the implementations secure a coherence to within the same order of magnitude as a self comparison of the simulation samples. The main advantage of the generative approach to simulating large volumes of cosmological structure is that much less time is required than creating N-body simulations on high performance computing resources. Future explorations of the method that are relevant to the next paradigm in cosmological simulations are also outlined. The most pertinent application being for a generator that can produce samples that maintain high statistical coherence to simulation data at various redshifts. At this stage, the generative artificial intelligence can then be said to be capable of generating an artificial universe.

1 Introduction

Beyond the distance scales of galactic diameters, toward the sizes of galaxy clusters and beyond, matter in the Universe appears as a webbed structure of filaments across volumes of emptiness known as voids. This network of luminous formation is known as the cosmic web [1–5]. The cosmic web as a distribution of matter contains artefacts linked to the state of the primordial universe, the nature of early universe perturbations and more fundamental physics to describe this era in cosmic time. The emergence of the structure is from an early period of exponential expansion, known as inflation [6]. This structure has been observed by missions such as [7, 8] and by weak gravitational lensing [9] through distortions, or shears [10], to images of distant galaxies caused by the large-scale structure of the universe.

The webbed structure is native to the well-established Λ Cold Dark Matter (ΛCDM) cosmology but its exact origins are not well understood [11] and at present it is believed that the structure originates from an amplification of quantum fluctuations as a density perturbation in the early universe [12], which is described by inflationary cosmology [13]. The matter distribution of webbed structure in the universe holds information on the true nature of dark matter, dark energy, the laws of gravity and the mode of galaxy formation [14–18].

Simulations that trace the evolution of phenomena such as structure formation, galaxy populations and the distribution of matter are imperative to our understandings of cosmological measurements. Our insight into the high-redshift deep field will increase with the amount of information provided by next generation observational missions such as [19–21]. As a consequence, so too does the need for higher resolution and more capacity in the volumes of simulated universes. Evidence for new theory in cosmology can come from greater observational resolution. This level of detail informs research into baryonic acoustic oscillations within the matter distribution [22–24]. Undoubtedly new information will emerge on these dilemmas in cosmology. This sets the next paradigm of the field in the study of smaller scales of cosmological structure.

Clearly there is a need for coherent simulations that are summarised by the same statistics as galaxy surveys such as the 2dF Galaxy Redshift Survery [8]. These simulations will then contain the same environments of large scale structure as observations to check experimental measurements with. Obtaining these more realistic simulations [25, 26] involves using greater computational resources and theoretical analysis. The latter of the two, due to the complexity of the non-linear physics that describes the growth of structure over cosmic time, shows some differences to observations [27]. A resource that could effectively reproduce the statistical properties of observations and simulations by learning the distribution in an advanced simulation (and therefore potentially of nature) with-

*Corresponding author: 2209131H@student.gla.ac.uk

out any assumptions of these properties would be of great value. This work aims to validate a method for obtaining a resource capable of achieving this objective by synthesising both 2D and 3D images from existing simulation data for two separate Generative Adversarial Networks (GANs) to emulate.

Deep learning has evolved to be able to process the statistical structure of the hierarchy of features in the visual domain [28]. There have been a number of recent applications of generative deep learning in astrophysics. GANs have been used to create new samples of galaxy images [29] and to obtain visual characteristics past the deconvolution limit [30]. Other applications use the GAN framework to generate 2D images of the cosmic web [31] and projected 2D mass distributions, known as convergence maps, from simulated data of weak gravitational lensing within the environment of an N-body simulation [32]. This report will show the insights that the study of astrophysics can have on AI through applications of statistics that could give new mechanisms for maximizing the realism of GAN generated samples.

This work is organised as follows. Section 2 will introduce the domain in which the methods for generating new galaxy-distribution samples originate. This section will introduce the sub-field of generative artificial intelligence and its relevance to the experimental method. Section 3 will describe Generative Adversarial Networks (GANs), their analytical properties and how these models learn to imitate datasets. Section 4 will study the present state of the concepts from the previous sections with respect to recent research in physics and astrophysics. Section 5 discusses some of the applications for successful potential of the work, that is, being able to generate distributions of galaxies that are coherent with N-body simulation data. Section 6 describes the methods of extracting and pre-processing training data from N-body simulations, the setup of each training loop for the GAN implementations and the architectures of the generative models. Section 7 gives insight into the methods of comparison between the generated and training distributions and finally Sections 8, 9 and 10 present the results, future work and conclusions.

2 Artificial intelligence

Artificial intelligence is the automation of a task that a human could know how to do themselves, that is, the automation of a classical algorithm using a machine. The computational power of a machine is used in a process that is instructed by a human programmer. An example of the function of an AI system was a game of chess between IBM's Deep Blue system and the grandmaster Garry Kasparov in 1997 [33]. This definition of AI includes machine learning (ML) and deep learning (DL) as well as more simple and mundane algorithms.

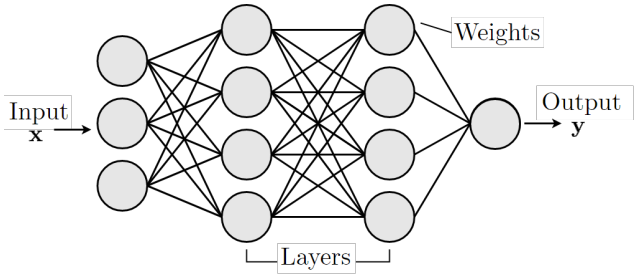

Figure 1: A basic neural network. The weights represent the parameters. The entire network represents the transformation $\mathbf{y} = f^*(\mathbf{x})$ that imitates the unknown function f of the equation $\mathbf{y} = f(\mathbf{x})$.

2.1 Machine Learning

Machine learning is the use of artificial intelligence without a pre-defined algorithm to derive the required output from the input given to a system [34]. A machine learning system is trained by a process of repeatedly comparing the performance of the system against an objective. These systems transform input data into meaningful outputs. The system is exposed to a collection of known examples of these inputs and outputs. The transformation from input to output in the system comes from the ability to learn representations of the input data that can be used to find deeper patterns in the input data [35]. Using computational resources in this way means that ML systems can go beyond the previous functions of tasks that are programmed by humans. Early examples of machine learning systems include speech recognition programs [36] and human world champion level performance at the game of backgammon [37].

Mitchell (1997) [38] defines the 'learning' of a machine learning system: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." A task T might be the de-blurring of a photograph or plotting the steps of a robot. The change in the performance P of a system is calculated in the training of the system and it is specific to the task T. An example is the classification accuracy of a model that classifies images or the mean squared error objective, where the predicted labels of images given to a model are compared to the true labels by calculating the square of their difference - both of these values could measure the performance of an image classification system. The experience E is a less well defined concept that depends on whether the system uses a supervised or unsupervised learning algorithm [39].

Machine learning consists of three types of learning problems that an algorithm is designed to solve: supervised learning, unsupervised learning and reinforcement learning (the latter is outside the scope of this report). Supervised learning represents situations in which the training data for the system is labelled with

its type in a dataset of objects. There are two main examples of supervised learning; the first is classification, where a class label is predicted from exposure to objects and their labels. The second is regression, where a numerical value is predicted from a dataset. Unsupervised learning algorithms process a dataset to learn the significant features and patterns in the dataset [40] and this is known as clustering. Density estimation is an unsupervised learning technique that estimates the function of a continuous field from a discrete set of points [41].

2.2 Deep Learning

The ideas of encoding, representation and compression in an ML system can be organised into a model that learns significant patterns and representations of input data. With the features of the data that are parameterised by the transformations that the system saves, the model can output an object that is dependent on the model parameters. This object could be a vector of probabilities representing the beliefs of a system on the identity or class of an image. It could also be an object with the same dimensions as the input data samples, created to imitate an object in the data distribution. A deep learning system organises these encodings by their similarity to principal components in the statistical structure of the data [35] using more advanced techniques of extracting data than typical ML systems.

The most common implementation of these ideas is through a neural network. Generally, with more insight being given in Section 3 on the most relevant models to this work, a neural network is a set of layers with parameterisations known as the 'weights' of the layer. The network is a composition of layered functions that approximate some function f that transforms an input $\mathbf{x}$ to an output $\mathbf{y}$ such that $\mathbf{y} = f^*(\mathbf{x}; \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ is the parameterisation of the layer that is learnt by the network. Figure 1 shows a basic neural network that transforms an input $\mathbf{x}$ into an output $\mathbf{y}$. The learning means that $f^* \to f$ and a strong representation of the transformation that was initially intractable is obtained. A further extension to deep learning systems from ML systems is the ability to learn non-linear functions of $\mathbf{x}$ by using a transformed input $\phi(\mathbf{x})$ with non-linear transform ϕ. The method of deep learning is to learn the transform ϕ using a model $y = f(\mathbf{x}; \boldsymbol{\theta}, \boldsymbol{w}) = \phi(\boldsymbol{x}; \boldsymbol{\theta})^T \boldsymbol{w}$. Here, $\boldsymbol{w}$ maps from $\phi(\mathbf{x})$ to the output of the model. This opens up much more generalisable performance than traditional machine learning algorithms [39].

This adaptability comes from arranging multiple layers of functions that learn increasingly meaningful representations. The depth of this hierarchy differentiates machine learning and deep learning, despite the fact that systems in both fields use similar bases. The mapping from input-to-output is through a one-way sequence of data transformations in the layers of the model architecture. These representations are derived from repeatedly exposing the system to data samples with feedback obtained from calculating the distance of the system from a desired objective. This feedback is calculated by differentiation of each functional layer with respect to its parameters whilst moving through the parameter space on the objective function for the model. This feedback is the result of the long standing backpropagation algorithm [42]. This algorithm computes the gradients of the objective function with respect to the model parameters. The backpropagation adjustment to the model parameters $\boldsymbol{\theta}$ is obtained from stochastic gradient descent (SGD) (see Appendix E.3.1) of the objective function in the parameter space of either model.

Figure 2: People that do not exist. Images created by NVIDIA's StyleGAN. Taken from [28].

This optimisation algorithm is required to find the optimal parameter values that give the best network performance in the parameter space. Typical image classification models may have $\sim 10^7$ parameters and above so this is not a simple task. This gives a high-dimensional space in which a global minimum exists for the objective function. The location of this point is specific to the hypothesis space for the task at hand.

Deep learning methods are used in the apparatus to give a framework for the ML architectures in the Generative Adversarial Networks for generating new cosmic web samples. These models are used to show the evolving differences between artificially generated data and simulation data. The next section explores these ideas.

2.3 Generative artificial intelligence

Artificial intelligence research has used neural networks to generate realistic data from existing examples and this is known as generative modelling. These new samples cannot be distinguished from samples drawn out of the probability distribution function of the data itself. The StyleGAN created at NVIDIA [28] is at the edge of modern generative AI. Despite the successes of these images being based on human-centred perceptions, the successes of recent research like this mean that reality can be the reference for the products of a generative system. This indicates the potential of generative AI for imitating complex data types. Figure 2 displays a sample of these artificially generated images.

Generative modelling with deep learning systems is used to learn the probability distribution that generated the original set of data presented to the system.

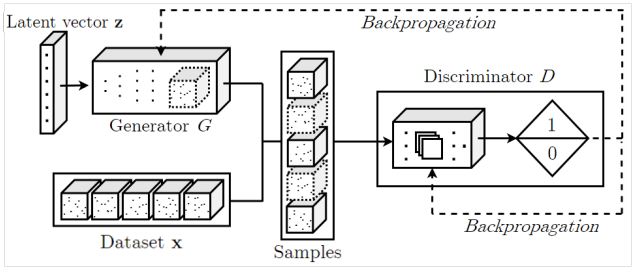

Figure 3: Schematic of the GAN framework; including discriminator and generator networks, the latent-space vector, dataset of samples and the backpropagation feedback to model parameters.

In a general sense this is the objective of the generative model. The result of training the generative model can be either or both of a probability density function or a program that can produce new samples from the probability distribution of the target data itself. This imitation is the ability to generate objects that could be said to come from the original target dataset.

Being able to generate objects, such as images, comes from creating a lower-dimensional basis of representations that summarise the statistical structure in the distribution of the objects. This basis is known as the latent space. The mapping from a point in the latent space to a realistic imitation of a real object is learnt and used by a generator model. Sampling points from the latent space enables the generative framework to create new samples of the original objects. The next section will characterise the generative model used in this work to generate galaxy distributions from simulation data.

3 Generative Adversarial Networks

The type of generative model chosen for the objective of this work was the Generative Adversarial Network (GAN) invented by Goodfellow et al. 2014 [43]. The successes of GANs are well established in a wide range of applications on natural images [44–46] and there are many extensions to GANs that open up a wide range of other successful models [47–49].

Generative Adversarial Networks consist of two models known as the generator and the discriminator. The generator must create samples that are allegedly drawn from the same distribution as the real samples dataset. The discriminator assigns scalars to a collection of input data and generated samples that represent the probability that each sample is real and from the data distribution p_{data} or fake and from the model distribution p_{model}. The discriminator has access to both real and fake samples. The training of a GAN involves navigating the combined parameter space of the generator and discriminator network parameters to find the optimal value of a function that gives the best performance of the networks at their opposing objectives. The metric that describes the performance is related to the distance between the model and data distributions. There is no need for a validation subset external to the training data because the generator does not have direct access to the training data. This is not the case for singular deep learning networks.

3.1 Generative adversarial learning

The discriminator and the generator are neural networks and each network tries to reduce their own objective function. This adversarial relationship is a game between the models rather than two optimisation problems. The optimum in this case is a Nash equilibrium [50] rather than a local minimum given by optimisation in the case of a singular neural network. In this equilibrium the returns of the objective functions only change by indirect influence of the other network and at this stage the gradients for backpropagation to either model have vanished.

The discriminator and generator models can be represented as functions D and G respectively because of their mappings as neural networks and parameter dependencies. The discriminator D has input $\mathbf{x}$ and parameters $\boldsymbol{\theta}^{(D)}$. The generator G takes a latent space vector $\mathbf{z}$ as input and uses parameters $\boldsymbol{\theta}^{(G)}$. Information flows through the network and its parameters to create the outputs $D(\mathbf{x})$ or $D(G(\mathbf{z}))$ for the discriminator and $G(\mathbf{z})$ for the generator. This is the forward-propagation from an input to an output. The model performances are calculated with the objective functions $J^{(D)}$ or $J^{(G)}$.

In the case of a GAN, the optimisation landscape is more complex because of the simultaneous learning of the two neural networks that both constantly adjust their parameters. The discriminator and generator models in a GAN depend on parameters $\boldsymbol{\theta}^{(D)}$ and $\boldsymbol{\theta}^{(G)}$ respectively. The models typically consist of multilayer perceptrons with the potential for other parametric components such as convolutional layers and the parameters are the state of these components at an epoch of training. The generator G draws noise vectors $\mathbf{z}$ from a prior distribution $p_{\mathbf{z}}$. The vectors $\mathbf{z}$ are passed through the generator architecture to create a sample $\mathbf{x} = G(\mathbf{z}; \boldsymbol{\theta}^{(G)})$ that is now a sample $\mathbf{x}$ from the model distribution p_{model}. The model distribution is the distribution of generated samples $G(\mathbf{z})$ obtained from passing latent vectors $\mathbf{z}$ through G such that $\mathbf{z} \sim p_{\mathbf{z}}$. The learning of the generator means that the samples evolve over the training process. Eventually, the samples will appear to have been drawn from the true data distribution p_{data} so that $G(\mathbf{z}) \sim p_{data}$. The generated samples $\mathbf{x} \sim p_{model}$ and real samples $\mathbf{x} \sim p_{data}$ become indistinguishable.

The mapping learnt by the generator is expressed as $G : \mathbf{z} \mapsto \mathbf{x}$. The discriminator D is a map from an input

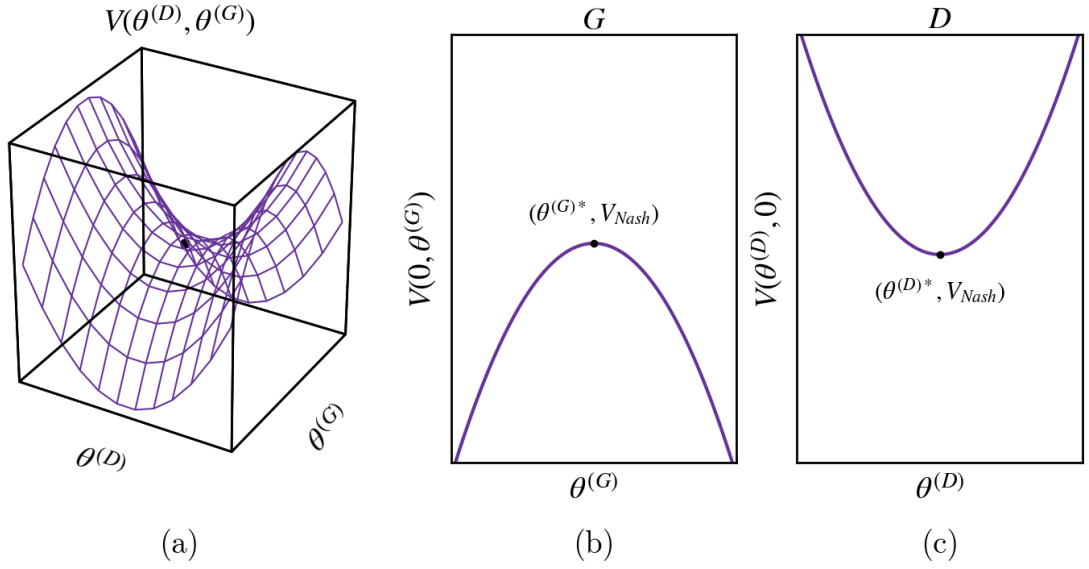

Figure 4: A diagram of the simplified two-parameter value function V for the zero-sum game in which the generator G and discriminator D participate in; (a) - the value function in parameter space with a Nash equilibrium point, (b) - The maximization objective for G with $\theta^{(D)} = 0$ and (c) - the minimization task for D with $\theta^{(G)} = 0$.

of real $\mathbf{x}$ and fake samples $G(\mathbf{z})$ to an output of as many probabilities. The values of $D(\mathbf{x})$ and $D(G(\mathbf{z}))$ signify the discriminator's confidence in the validity of the sample. This mapping is expressed as $D : \mathbf{x} \mapsto [0, 1]$.

3.1.1 Formulation of GAN learning

The objective functions of D and G are defined in terms of both players' parameters and are denoted by $J^{(D)}(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)})$ and $J^{(G)}(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)})$ respectively. In the most basic adversarial setting for the GAN models, D and G play a zero-sum game where $J^{(D)} = -J^{(G)}$.

During training the discriminator minimises $J^{(D)}$ only through feedback from backpropagation to its parameters $\boldsymbol{\theta}^{(D)}$. The generator G minimises $J^{(G)}$ by only affecting its parameters $\boldsymbol{\theta}^{(G)}$ through returns from backpropagation related to the discriminator classifications. The minimization objectives are simultaneous between two networks in search of a Nash equilibrium and added together the objectives for the discriminator and generator give a value function $V(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)})$ in the combined parameter space $(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)})$. Figure 4 shows the value function in the combined parameter space for which the backpropagation gradients are calculated with D and G as adversaries of each other. The optimum parameters for each network are shown at the Nash equilibrium. The game between the models is described with the value function V as

$$\begin{aligned} \min_G \max_D V(D, G) &= \mathbb{E}_{\mathbf{x} \sim p_{data}} \log D(\mathbf{x}) \\ &+ \mathbb{E}_{G(\mathbf{z}) \sim p_{model}} \log(1 - D(G(\mathbf{z}))) \end{aligned} \tag{3.1}$$

in the combined parameter space. This equation shows the opposition between the maximization of the correct classifications by D and the minimisation of the classifications by D of the generator products that are made by G.

The Nash equilibrium is satisfied by a pair of parameter sets $(\boldsymbol{\theta}^{(D)*}, \boldsymbol{\theta}^{(G)*})$ for which no further changes occur to the objective functions $J^{(D)}$ and $J^{(G)}$ of both networks [50]. Equation 3.1 means that the zero-sum game for the models returns an adjustment $+V$ for the discriminator and $-V$ for the generator. Figure 4 shows the equilibrium point in the parameter space and the returns $V(\boldsymbol{\theta}^{(D)}, 0)$ and $V(0, \boldsymbol{\theta}^{(G)})$ for the discriminator and generator with optimum parameters $\boldsymbol{\theta}^{(D)*}$ and $\boldsymbol{\theta}^{(G)*}$ respectively in the zero-sum game. The zero-sum game between D and G defines an objective function $J^{(D)}$ for the discriminator

$$\begin{aligned} J^{(D)} &= \mathbb{E}_{\mathbf{x} \sim p_{data}} \log D(\mathbf{x}) \\ &+ \mathbb{E}_{G(\mathbf{z}) \sim p_{model}} \log(1 - D(G(\mathbf{z}))) \end{aligned} \tag{3.2}$$

this is from the cross-entropy discriminator objective function for the binary classification of real and fake samples. Appendix C.3 defines cross-entropy in this context. The expectation operators $\mathbb{E}_{\mathbf{x} \sim p_{data}}$ and $\mathbb{E}_{G(\mathbf{z}) \sim p_{model}}$ correspond to the ensemble averages over the classifications of real and fake samples respectively. Equation 3.2 shows that the first term cannot be influenced by the generator G because here the samples are drawn from the data distribution to which it has no access to. Inputs to the generator $\mathbf{z}$ are drawn from the prior distribution $p_{\mathbf{z}}$ over the latent-space to the model. The discriminator is given a generator sample $G(\mathbf{z})$ to classify it as fake, so $D(G(\mathbf{z})) \to 0$ whereas the generator G tries to make $D(G(\mathbf{z})) \to 1$. The Nash

equilibrium is satisfied for the model and data distributions becoming identical (in the ideal case) so that $G(\mathbf{z})$ is drawn from p_{data}. This means the discriminator classification tends to $D(\mathbf{x}) = \frac{1}{2}$ for all samples $\mathbf{x}$ [43]. The optimum of the generator G parameters $\boldsymbol{\theta}^{(G)*}$ is

$$\boldsymbol{\theta}^{(G)*} = \underset{\theta^{(G)}}{\operatorname{argmin}} \max_{\theta^{(D)}} V(\theta^{(D)}, \theta^{(G)}) \tag{3.3}$$

This motivates the generator to increase the likelihood of the discriminator incorrectly classifying a sample. The discriminator tries to increase its own likelihood of correctly classifying a sample by minimising the binary cross-entropy objective. The local minima of the value function V are not necessarily Nash equilibrium points, rather they are points that simultaneously minimise the objective functions $J^{(D)}$ and $J^{(G)}$. Whilst both networks search the parameter space via SGD of the value function, there is no guarantee of a Nash equilibrium, this is part of what makes training an adversarial network a difficult task.

The cross-entropy for the discriminator classification of a sample against the true label of the sample is numerically effective for a GAN in practice because this objective does not saturate for an incorrect identification. Because the networks try to minimize their opposing objectives, if the discriminators performance increases enough, the generator objective function tends to zero. This causes the generator performance to become static. At all points in the parameter space the generator objective minimization is guided by the gradients from backpropagation which will reduce to zero. At this point the discriminator cannot improve and less information is back-propagated through the networks.

The solution, shown in [43], is to deviate from the zero-sum ideal by inverting the target used to create the cross-entropy objective function for the generator. The generator's objective function becomes

$$J^{(G)} = \mathbb{E}_{G(\mathbf{z})\sim p_{model}} \log D(G(\mathbf{z})) \tag{3.4}$$

Now the generator tries to maximize the log-probability of the discriminator wrongly classifying a given sample. This objective with the original objective for the discriminator $J^{(D)}$ means there are always non-vanishing gradients for backpropagation despite any imbalance in discriminator and generator performance.

3.2 Experimental design of GANs

The opposing players in the zero-sum game are assumed by two deep convolutional neural network (DCNN) models. These networks use convolutional layers to extract features from grid-like data. Deep Convolutional Generative Adversarial Networks (DCGANs), first implemented by Radford et al. 2015 [51], use these layers to create images. The DCGAN model has shown excellent results on natural image generation using the CelebA [52] and the LSUN-Bedrooms [53] datasets. The DCNNs in this work are based on the AlexNet models for image classification [54]. Despite most of the architectural features, such as convolutional layers, being used in deep learning models prior to this publication the DCGAN architecture improved previous GAN architectures for increased performance in image generation tasks. See Appendix E for details on the relevant tools used in DCGANs. The models in the methods of this work borrow from the ideas of this architectural basis.

3.3 Variational Auto-Encoders

Generative artificial intelligence is able to create realistic natural images sampled from a latent space, but different models are able to perform these tasks. The distinctions between them explain the choice of the GAN in the experimental objective. The most prevalent models in the literature are GANs and Variational Auto-Encoders (VAEs) [55, 56]. VAEs are known [50] to be well-posed to learn latent spaces that are structured, where specific directions encode strong variation in the data whereas GANs, whilst being able to generate realistic images, draw from a latent space that has less structure and continuity [35]. It is likely that the structure of the cosmic web is dependent on factors that are not as immediately present in the data compared to dependent structure that is more obvious natural image data. For this reason Generative Adversarial Networks were chosen for this work. The VAE model consists of encoding and decoding modules. An object is encoded into the parameters of a statistical distribution - thus assuming it is from a distribution. A point is sampled from the latent distribution that generates an input object. The decoding module maps this point in the latent space back to the original input object - this is the alleged reconstruction of a new object from the original distribution. The equivalent to the generator model in the VAE framework is the decoder. In training the VAE has two objective functions; a reconstruction loss that ensures the decoded samples match the initial inputs and a regularization loss that reduces over-fitting to the training data samples.

4 Related work

Over the last few years research has accumulated on the different applications of GANs to emulating data samples. An example of GANs being able to provide new data for further statistical insight is in the domain of particle physics. Here, GANs have been used to generate particle beams [57], particle interaction-showers [58] and 2D images of jets [59]. The DCGAN architecture has consistently shown that it could generalise well in astrophysical applications. The ability for GANs to transfer to new data and generalise to different types of data has been founded by many publications with a variety of target data.

Aside from the research into GANs for image based tasks, less work exists that uses generative AI in astrophysics. One application of GANs to astrophysical data concerns subverting the point spread function (PSF) that describes the effect of noise on the clarity of images [30]. Typically the deconvolution of the separate noise and true light distributions are limited by the Nyquist sampling effect. The findings indicate the GAN approach successfully obtains artefacts within images that have low signal-to-noise ratios. See [29] for a similar example of research using GANs in generating astronomical images.

Rodriguez et al. 2018 [31] use GANs to generate realistic 2D samples of N-body simulation mass distributions of size 100 h^{-1} Mpc and 500 h^{-1} Mpc. This paper is the closest to the work presented in this report. N-body simulation data was obtained from the particle distribution of the L-PICOLA simulation [60] at $z = 0$. This simulation is based on the ΛCDM cosmological model with the Hubble constant $H_0 = 70$ km s^{-1} Mpc^{-1}, dark energy density $\Omega_\Lambda = 0.72$ and matter density $\Omega_m = 0.28$. The method of Rodriguez et al. 2018 differs from this report in the statistical tests of the generator output. The generated samples are cross-correlated with each other in pairs to quantify their mutual independence. This estimates the effect of a weakness of GANs against other generative models, known as mode collapse [43], where generated samples can often be very similar if not identical. The generated 2D slices are compared to simulation samples in a cosmological context by calculating the automatic and cross (sample-to-sample) power spectrum of the 2D images using a discrete Fourier transform. The power spectra P_k (see Appendix A) of the generated samples against the training samples agree to 1-2% for the majority of k values. This work shows the potential for GAN methods to subvert N-body simulations.

Work by Mustafa et al. [32] uses a GAN to generate weak lensing convergence maps. The lensing maps are generated from GADGET2 N-body simulation code [61] that are then ray-traced using Inspector Gadget weak lensing simulation procedures [62–64] to produce a collection of ray-traced lensing maps at $z = 1.0$. The viability of replacing simulations with generative models is discussed, with detail on the nature of the separation between the model and data distributions after training. This is quantified by measuring the statistical confidence in the null hypothesis; the statement that the summary statistics in the generated maps and the validation maps originate from the same distribution. The summary statistics are the pixel intensities, the power spectrum of a map and a non-Gaussian statistic pertaining to the nature of the structure in a given map. The Kolmogorov-Smirnov (KS) test for the pixel intensities of the generated maps gives a p-value of $p_{\mathrm{KS}} > 0.999$ in comparison to the real data. The generation of completely new maps that are distinct from the training set is verified. The referenced research shows the strengths of GANs in generalising to new data and in emulating the distributions that are native to different cosmological datasets.

5 Applications of an artificial universe

5.1 Cosmological simulations

The Millennium Project simulation [65] and the successive Millennium-XXL [26] and Millennium-II [25] simulations made foundations for the comparison of observation to theory-based simulation. Research using the simulations has provided new insight into high-redshift quiescent galaxies [66], dark-matter haloes [67], weak lensing [68], galaxy clustering [69] and the direct comparison of galaxy surveys with galaxy groups predicted in the simulation volumes [70]. These applications in the GAN-generated galaxy distributions depend on extending the depth of the method presented in this report but they are strong motivations for the use of GANs in simulating cosmological data. The direct comparison of survey data and generated data is the more likely once the objective of this work is completed. The approach in this report can generate a similar volume to the Millennium simulation in around an hour (GPU time) as opposed to three-hundred years (CPU time) for the Millennium simulation.[1].

5.2 Gravitational wave sources

Measurements of cosmological parameters from observations of gravitational wave events are independent of the electromagnetic (EM) spectrum and any obstacles to the transmission of EM radiation. Because these measurements do not use EM radiation, they provide alternative estimates that do not depend on cosmological distance scale [71]. Information on dark energy density at $z > 0.1$, past the edge of the local universe, could be obtained to better constrain cosmological parameters. At this distance, the cosmological distance-redshift relation depends on more than just H_0 [72]. A further dependency is on the matter and dark energy densities Ω_m and Ω_Λ. The values of these parameters could be constrained by measurements of the background rate of expansion from gravitational wave (GW) events at non-local redshifts. Next generation GW detectors such as Cosmic Explorer [73] and the Einstein Telescope [74] as well as space based detectors such as LISA [75] will be at the forefront of such observations. There are also gravitational waves from epochs in the early universe. These are known as relic gravitational waves and they can be attributed to GWs in the paradigm of quintessential inflation [76]. GW astrophysics has potential in the future to open new windows onto the conditions of the early universe.

[1] wwmpa.mpa-garching.mpg.de/galform/virgo/millennium

Gravitational waves were first observed in 2015 with the detection of the binary black hole merger GW150914 [77]. A separate GW observation was of the binary neutron star merger event GW170817 [78] and later the optical redshift measurement of its electromagnetic counterpart, the host galaxy NGC4993 [79, 80]. This event was used to constrain the value of the Hubble constant H_0. The value of H_0 was found to be $H_0 = 70^{+12}_{-8}$ km s^{-1} Mpc^{-1}. This was the first application of gravitational wave sirens in obtaining cosmological parameters such as H_0. A separate analysis that improved on the X-ray and radio emission modelling from GW170817 [81] obtained a value $H_0 = 74.0^{+11.5}_{-7.5}$ km s^{-1} Mpc^{-1}. The consensus for the true value of H_0 from electromagnetic spectrum observations is a contentious issue. Assuming the ΛCDM cosmology, the Planck experiment found a value of $H_0 = 67.4 \pm 0.5$ km s^{-1} Mpc^{-1} [82].

The same event GW170817 had a degeneracy between the source distance and the viewing angle to the radio counterpart that limited the initial certainty in the EM localization of the GW source. This degeneracy was broken from radio interferometry [83] and a value of H_0 that disagrees with the previous values was found to be $68.9^{+4.7}_{-4.6}$ km s^{-1} Mpc^{-1} [84]. A recent survey calibrates Type Ia supernovae (SNe Ia) against a set of Tip of the Red Giant Branch (TRGB) stars [85]. These are bright stars that have just initiated helium burning in their cores and H_0 from this calibration was found to be $69.8 \pm 0.8\,(\pm 1.1\%\ \text{stat}) \pm 1.7\,(\pm 2.4\%\ \text{sys})$. The disagreement between the competing values of H_0 could be decided using more measurements from GW event observations and it is expected that accuracy similar to the level of measurements by Planck is possible if between 10^6 and 10^7 gravitational wave events are observed [86].

Gray et al. 2019 [87] consider GW observations where the electromagnetic counterpart to the GW source is unobservable so the distance to the electromagnetic counterpart is inferred as well as the GW source distance. This may become the norm for future searches into narrowing the uncertainties on cosmological parameters. The paper describes Bayesian methods used to obtain posteriors on H_0 from GW events.

The process involves the creation of a series of mock data challenges (MDCs). The MDCs are simulations of GW events in mock galaxy catalogues. Between the separate MDCs there are varying conditions of GW and EM selection effects (GW selection effects correspond to detector sensitivity and EM effects are due to observed flux limitations). The effects of clustering and large scale cosmological structure are ignored in the simulation of co-moving volumes of uniformly distributed galaxies that the GW events occur in. These mock catalogs show different comparisons to certain properties of true galaxy distributions from surveys for locating GW events, such as GLADE [88]. A larger testing volume could be generated from a routine, such as the one described in this report, with the summary statistics that are similar to survey or simulation data with added realistic cosmological structure. The EM information from outside the volume specified in the catalogs could not be replaced by the GAN method shown in this report because at present it only returns positional data. This could increase the understanding of how current techniques for observed GW events perform so that tighter constraints on the source-observer separation could be obtained in the future. This would help to accumulate more data for posteriors to further constrain H_0.

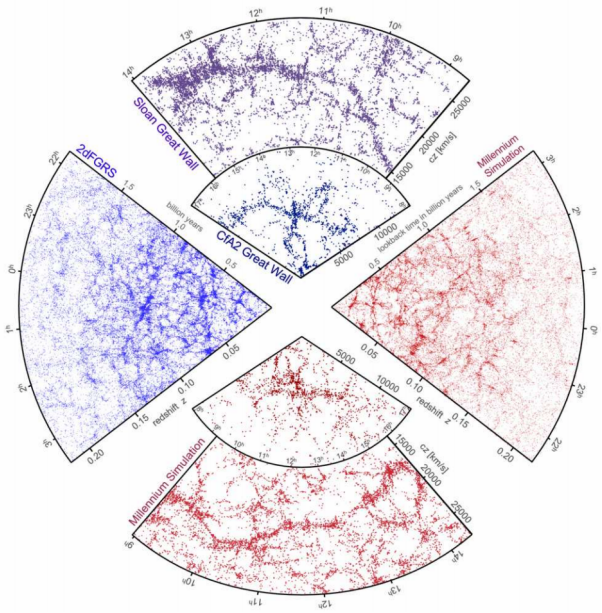

Figure 5: A comparative diagram of Millennium simulation data against galaxy survey data[2].

In short, expanding the range for credible inferences of GW event localizations with increased understanding of GW event analyses would provide a more accurate estimate on the value of H_0 and other important parameters. This will be important in the next runs of LIGO [89]. The completed runs have already shown the higher frequency of binary black holes found at larger distances compared to binary neutron stars [90], where current galaxy catalogs are incomplete. The true substitute of the incompleteness could be replaced by a fast and realistic simulation from a conditional generative model that is given a redshift value. It should be noted that there is no reason such a generative model could not generalise to higher redshift simulation data for this purpose. This is left to the discussion of future work in Section 9. This idea shows the direct link between generative AI and new discoveries in astrophysics.

[2]Image from wwmpa.mpa-garching.mpg.de/millennium.

6 Methods

6.1 Simulation data

Hydrodynamical simulations of particles with masses on the order of a typical galaxy are the best way to emulate cosmological phenomena for testing without observational complications. Cosmological simulations model the physics in the evolution of baryonic and dark-matter in a self-consistent way that does not depend on observations.

The Millennium Project Simulation was used to sample training data for the generative models. The Millennium simulation was published in 2005 by Springel et al. [65] and remains active in research today. The simulation was run using the GADGET2 code [61] and it contains 2160^3 particles of mass $8.6 \times 10^8\, h^{-1}\, M_{Sun}$ in a box of side length $512\, h^{-1}$ Mpc. The cosmological parameters of the Millennium simulation are based on WMAP-1 data [91] and the 2dF Galaxy Redshift Survey [8]. These are $\Omega_m = 0.25$, $\Omega_\Lambda = 0.75$, $\Omega_b = 0.045$ and $h = 0.73$. Figure 5 shows a comparison of the Millennium simulation to the large scale structure and redshifts of the 2dF Galaxy Redshift Survey.

The galaxy catalogs containing physical properties of galaxies in the Millennium simulation were derived with the Semi-Analytic Galaxy Evolution (SAGE) codebase by Croton et al. 2016 [92] that improves on the earlier model of Croton et al. 2006 [93]. The SAGE catalogs were sourced from the Theoretical Astrophysical Observatory (TAO) [94]. SAGE modelling adds baryonic processes into N-body simulations of halos after the simulations have been run. The baryonic particle properties are from information in the simulation run such as mass, size, spin and their merger history. SAGE models account for different mechanisms that affect the evolutions of galaxies and therefore the evolution of cosmological structure. The N-body particle positions at a redshift $z = 0$ snapshot of the Millennium simulation gave the positional training data for the GAN.

6.2 Extraction of training data

Position vector components for the centres of cube samples in the simulation volume were drawn from a uniform distribution along each axis. If the addition or subtraction of half the side length of the cube to any of the components resulted in a point outside of the simulation volume, the point was discarded. The galaxy cube samples were obtained by sorting the position vectors of galaxies in the simulation that were within the cube volume. Each point inside the cube was recorded in an array. From the simulation volume of $512\, h^{-1}$ Mpc an ensemble of 128 array samples of size $80\, h^{-1}$ Mpc were selected. The galaxy positions in each array for every cube were scaled to the range $[-1, +1]$. 3D-histograms with 32 bins per axis were made on each cube across the same range for each axis.

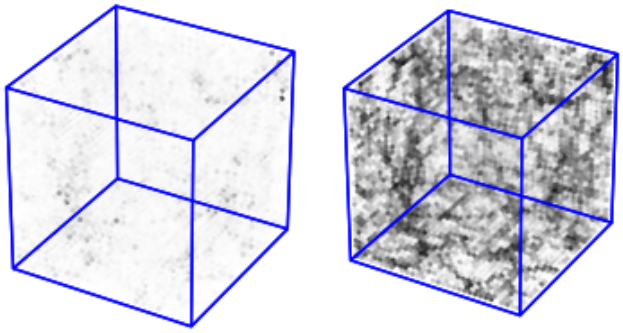

Figure 6: The application of the scaling s to a 3D-histogram training sample. The cube on the left shows an image x and on the right is a scaled image $s(x)$.

The densities in each bin cell were also scaled to this range to fit the output ranges of the GAN models. For the 2D-samples, projections perpendicular to the xy-plane were taken for each of the collected cubes. Here a projection means ignoring the z-component of every point in the 3D-sample to give a 2D-sample as if one was facing the xy-plane and looking into the cube. Two-dimensional histograms were made on these projections to create a collection of images with 32 bins along each axis. After testing different sizes, a cube size of $80\, h^{-1}$ Mpc was found to balance the selection of unique samples in the simulation volume against featuring the same features in the simulation structure.

6.3 Initial data transformations

The 3D-histograms have pixel densities that span across many orders of magnitude from voids to dense clusters of galaxies. The origin of GANs from image generation tasks has optimised them for the general statistical structure of these images - the pixel densities in natural images are typically more uniformly distributed over the same scales. To help the GAN in generating sparsely distributed densities a transform, first utilised in [31], was used on the histograms to enhance the contrast between the density features. The transform from the original histogram x to the scaled histogram $s(x)$ is

$$s(x) = \frac{2x}{x + a} - 1 \tag{6.1}$$

where a is a free parameter that depends on the sparsity of the histogram densities. The value of $a = 10$ was found to be optimal for the $80\, h^{-1}$ Mpc histograms. This parameter remains fixed during training. The value of a controls the median pixel density of the scaled image. Figure 6 shows the difference between scaled images $s(x)$ and the original images x. This transform was not necessary for the 2D-GAN implementation. The inverse transformation $s^{-1}(x)$ can be applied to obtain the original image. A logarithmic scaling was used originally, but this did not provide good results with the GAN architecture that used the

hyperbolic tangent (tanh) activation because the logarithm of a negative number is undefined. The models using tanh activations achieved much better results than models without these activations.

6.4 Networks, architectures and training

6.4.1 2D-GAN implementation

Architecture — The networks follow the structuring of the DCGAN models [51] and are written in Keras [95]. A full description of the layers in the 2D-GAN architecture is given in Table 1 of Appendix B. The generator transforms a 100-dimensional vector drawn from a Gaussian prior $\mathbf{z} \sim \mathcal{N}(0, 1)$ into a 32×32 single-channel image. The latent-space vector is followed by a fully connected layer that is reshaped to a stack of feature maps for the two convolutional layers. Batch-normalisation is only used after the fully connected layer. Through each layer hyperbolic tangent activations are used. See Appendices E.2 and E.3 for more detail on batch-normalisation and activation functions.

The discriminator network has two convolutional layers and two fully connected layers. The image samples have features extracted by the first two convolutional layers and the features are flattened for the final fully connected layers. Hyperbolic tangent functions are used throughout the model and a sigmoid activation function (see Appendix E.2) is used for the final single neuron layer that outputs the probability for a given sample being real.

Both networks use the same number of filters for the corresponding layers in either model. Together the discriminator and generator models have 3.8×10^7 trainable parameters. The discriminator and generator objectives of Equations 3.2 and 3.4 are minimized using the Adam optimiser [96]. Each model uses a learning rate of 2×10^{-4} and momentum $\beta_1 = 0.5$. The models use a batch size of 64 histograms out of a set of 128 training histograms, where the batch size is the number of samples given to the discriminator at each iteration of training. Refer to Table 2 of Appendix B for a complete listing of the 2D-GAN hyper-parameters used in each model. The hyper-parameters of a model are set outside of training and they shape how the training proceeds.

Training — GANs are notoriously difficult to train. This means that the state of the models can generate realistic samples for a single epoch at a time. The GAN was made to record the parameters of the generator model at these epochs by using a statistical check (see Section 7) of the generated histograms against the training data histograms in the training loop. The loss (the objective function value) and accuracy values of the discriminator for each epoch were recorded during training.

An indicator for the performance of a GAN is the classification accuracy of the discriminator model. This metric is the average classification accuracy of the discriminator on two separate batches of real and fake samples. The accuracy metric for the generator is given by the same accuracy subtracted from one. However, in this case the accuracy is the discriminator's accuracy in classifying a batch of fake cubes labelled as real. This is because the generator is trying to make the discriminator classify its products as real. The best results were obtained when the losses of the networks diverged and the accuracies of the networks slowly increased and were at approximately the same value. A training loop of 3.0×10^4 epochs for the 80 h^{-1} Mpc cubes sectioned into 32×32 pixel projections took approximately 2 hours on an NVIDIA Tesla V100 GPU with 32GB capacity.

6.4.2 3D-GAN implementation

Architecture — Table 3 of Appendix B shows the architectures of the 3D-GAN discriminator and generator models. The hyper-parameters used in the 3D-GAN are also given in Table 4 of Appendix B. Together the discriminator and generator models had around 2.5×10^8 trainable parameters across one linear layer and three convolutional layers for the generator model and three convolutional layers for the discriminator model.

Training — The 3D-GAN implementation was limited by the GPU memory capacity - the parameters and architectures together with the sample batch sizes meant that the GPU was easily saturated. Deeper models with more convolutional units were implemented but these tended to stall the GPU. A training loop for 8.0×10^4 epochs for the 80 h^{-1} Mpc cubes formatted to $32 \times 32 \times 32$ pixel histograms took approximately 72 hours on an NVIDIA Tesla V100 GPU with 32GB capacity.

Generative adversarial networks are harder to train with three-dimensional input data because of the increase in the number of configurations of interest in the data. There are a number of tricks and techniques for moving toward more convergent and stable training. A compilation of these resulted from testing GANs in natural image tasks. The techniques that were used to train the 3D-GAN in this work were one-sided label smoothing, historical averaging and a decaying noise input which were borrowed from [97, 98]. The exact implementations stem only from the original theoretical basis. One-sided label smoothing adds scalar values sampled from a uniform distribution $\mathcal{U}(-0.1, 0)$ to the real labels only. Historical averaging adds a penalty L_H to the objective functions of the discriminator and generator. The addition is

$$L_H = \left|\left| \boldsymbol{\theta} - \frac{1}{t} \sum_{i}^{t} \boldsymbol{\theta}[i] \right|\right|^2 \tag{6.2}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\theta}[i]$ are the model parameters at the present and previous epochs respectively. This cost term helps the networks move out of mode collapse. The decaying noise input temporarily slows the discriminator training by showing it less clear real samples. This in turn allows a strong discriminator model that doesn't overwhelm the generator early in training. The decaying noise input samples noise at each epoch to add to every histogram pixel in an ensemble of samples. The Gaussian noise $\mathbf{n}$ has standard deviation σ_i that depends on epoch i such that

$$\mathbf{n} \sim \mathcal{N}(0, \sigma_i), \quad \sigma_i = A\mathrm{e}^{-\gamma i} \tag{6.3}$$

where the free parameters $A = 10^{-2}$ and $\gamma = 10^{-3}$ were found to work well. These techniques were not necessary in the 2D-GAN implementation. This is likely due to the relative information content between the 2D and 3D samples. Diluting the real and fake batches of samples to 1% of the batch-size with the opposite validity of samples helped the training process stabilise.

7 Statistics and diagnostics

7.1 Comparison of distributions with Kullback-Leibler Divergences

Generative Adversarial Networks reduce the difference of the model probability distribution between the generated samples p_{model} and the probability distribution of the training data samples p_{data}. This reduction quantifies how well p_{model} estimates p_{data}. An estimate on the suitability of this replacement was made using a comparison with different ensembles of samples. The ensembles were collected from the training, generated and uniform distributions of histograms denoted t, g and u. The uniform distribution u was a set of histograms with pixel densities drawn from a uniform distribution $\mathcal{U}(-1, 1)$. The comparison of sample ensembles tested how well the generator made the discriminator associate the real and generated samples to the same distribution. The Kullback-Leibler (KL) divergence and its relevance to the comparisons is defined in Appendix C.

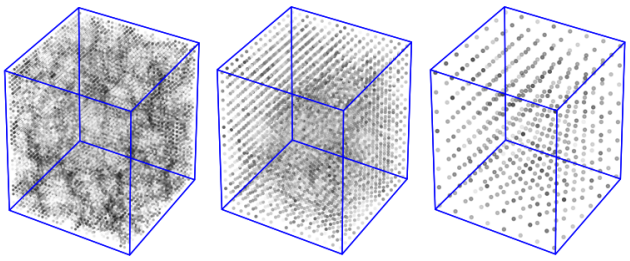

Figure 7: A histogram down-sampled with three kernels of size $k = \{1, 2, 4\}$ (from left to right).

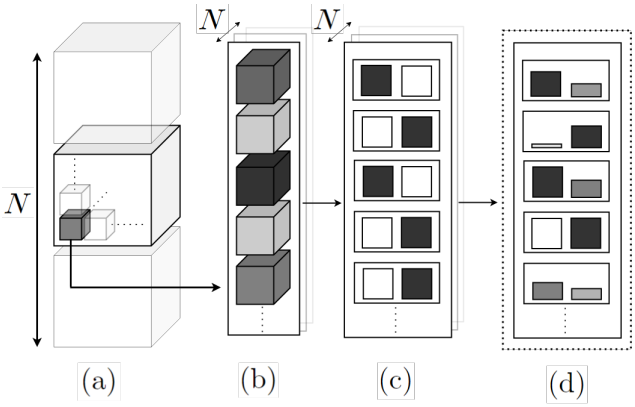

Figure 8: From a single ensemble of 3D-histograms to a normalised distribution; (a) - an ensemble of 3D-histograms that will be transformed into a distribution, (b) - the flattened vector of bin-cells, (c) - the histograms of each vector-component, (d) - the final collapsed vector encoding the whole of a given distribution of 3D-histograms.

7.1.1 Histogram distributions

After selecting the best epochs from training, distributions of training and generated samples were gathered and the average relative entropy between the ensembles was calculated to estimate how similar the generated and simulation samples were to each other. The uniform distribution u was made to have samples to compare with that did not have any correlated structure on any scale. The comparison of ensembles shows the spread of the D_{KL} value on samples from the original distributions t, g and u and consequently how different random groups of samples are to each other. In the case of the uniform-training and uniform-generated comparisons it shows how similar the samples are to a uniform distribution. The D_{KL} values between draws of samples from distributions a and b were calculated for both $D_{KL}(a||b)$ and $D_{KL}(b||a)$.

7.1.2 Deriving the KL comparative diagnostic

There were 50 samples in the ensembles from the distributions t, g and u that contained 500 samples each. Each histogram in the ensemble was down-sampled by iterating a kernel of size k over each histogram and summing the densities. The summed densities were divided by the number of bins in the kernel after down-sampling. This is equivalent to the initial histograms having a lower sampling with larger bins. After an ensemble is down-sampled, the ensembles of two distributions are compared. The samples are individually flattened to 'histogram-vectors' with $(32/k)^3$ components in the 3D case and $(32/k)^2$ in the 2D case. This is shown in Figure 7. One-dimensional histograms with two bins were made for each component in the histogram-vectors. The two bins represented 'high' and 'low' pixel densities and the boundary between them

was the median of the training data pixel densities. This is because using the median meant that it was not assumed that the pixel density distributions were symmetric. The histogram-vectors were then collapsed over the ensemble and normalised to sum to one as a probability distribution. The vector was then sampling the probability density function of pixel density. This process is illustrated in Figure 8.

The distributions were compared with all the possible pairings of the set the distributions $\{t, g, u\}$. The relative entropy, that is equivalent here to the KL divergence D_{KL}, was calculated between the distribution ensembles. The process was repeated for 100 separate ensembles and with the final D_{KL} values logarithmically-scaled D_{KL} histograms for each comparison were made. This process was repeated for each kernel size k where $k \in \{1, 2, 4\}$. The comparisons were the same for the 3D case and the 2D case.

8 Results

8.1 Generated images

8.1.1 2D-GAN images

The results from the separate implementations were the generated images, the model loss and accuracy curves, the training statistic curves and the ensemble D_{KL} trials. A collection of training and generated samples from the 2D-GAN implementation are shown in Figure 9. The generated 2D-histograms are very similar in appearance to the training data histograms. This is the main reason for the tests of statistical structure in ensembles from either distribution. The 2D-GAN images show most of the important features of structure such as filaments, voids and dense clusters.

8.1.2 3D-GAN images

A collection of 3D-histograms generated through the 3D-GAN implementation are shown in Figure 10. Two-dimensional projections of these same samples are shown in Figure 11. These projections can be compared to the 2D-GAN generated images because they are derived in the same way from the same data, though the scaling $s(x)$ is shown on the 3D projection images.

The 3D images in Figure 10 show that the values of the pixel densities in the generated histograms are similar to the training data. This is the easiest characteristic of the images to judge because examples of structure in these images are sampled from the distribution of all its different possible appearances. The larger scale filaments and sheets in the generated cubes are not always reproduced but they are not totally missing from the histograms. Closer inspection in an interactive environment shows realistic voids in the histograms.

The 2D-projections of the histograms show the same agreement in the pixel densities between the training and generated samples. The main difference is in the 'blur' effect on the generated projections. This could be solved by changing the kernel size and initialization in the convolutional layers of the models. The projections also show more clustering and some larger filament structure relative to the 3D-histograms. The smaller scale variation of the pixel density in the training data appears more random than the generated data if one picks a line of twenty or so pixels in an image. It could be beneficial for the GAN models to use a higher pixel count in the training data so that less of the information that records a given feature is lost in fewer pixels. The most important statistic of the generated images to test is a cosmological characteristic of the data such as the power spectrum P_k or the correlation function ξ. This is left to the discussion of future work in Section 9 for now.

8.2 Loss and D_{KL} curves

8.2.1 2D-GAN implementation

Figure 12 shows the 2D-GAN objectives and accuracies over a run of training. The behaviours of the models over the training are apparent in these curves. The sharp jump at the start over the first few 100 epochs or so shows the initial learning for both models that are new to their tasks. The jagged shape over next 1.5×10^4 epochs in the accuracy curve (as well as the loss curve) is where the generator fools the discriminator with half of its generated samples. The discriminator, at the same time, is only able to correctly classify half of its input of real and fake samples. There are small fluctuations in the accuracies of either model as the generator makes smaller adjustments relative to the initial training, until just after 1.5×10^4 epochs. There is a sharp jump in the losses of both models, that is highest for the discriminator, which implies the generator attempted a new strategy for its objective. The accuracies here both climb to 100% whilst the losses decrease to a lower value. At this point the generator cannot do any better. This does not mean an ideal 'perfect' generator has closed the distance between the model and data distributions, but instead that the architecture of the discriminator cannot extract any more useful information (and is extracting less and less) to the generator. The gradients for backpropagation have vanished. This was seen in training. The generator suffered from mode collapse and made much more similar samples.

Figure 13 shows the KL divergences and Kolmogorov-Smirnov (KS) p-values p_{KS} for the generated and training image comparisons over a run of training. The KS statistic and its p-value are a sensitive test between two distributions. The KS statistic D_{KS} and the p-value p_{KS} are defined in Appendix D. This test is useful in monitoring the performance of the GAN, particularly near convergence where the gradient adjustments to the models

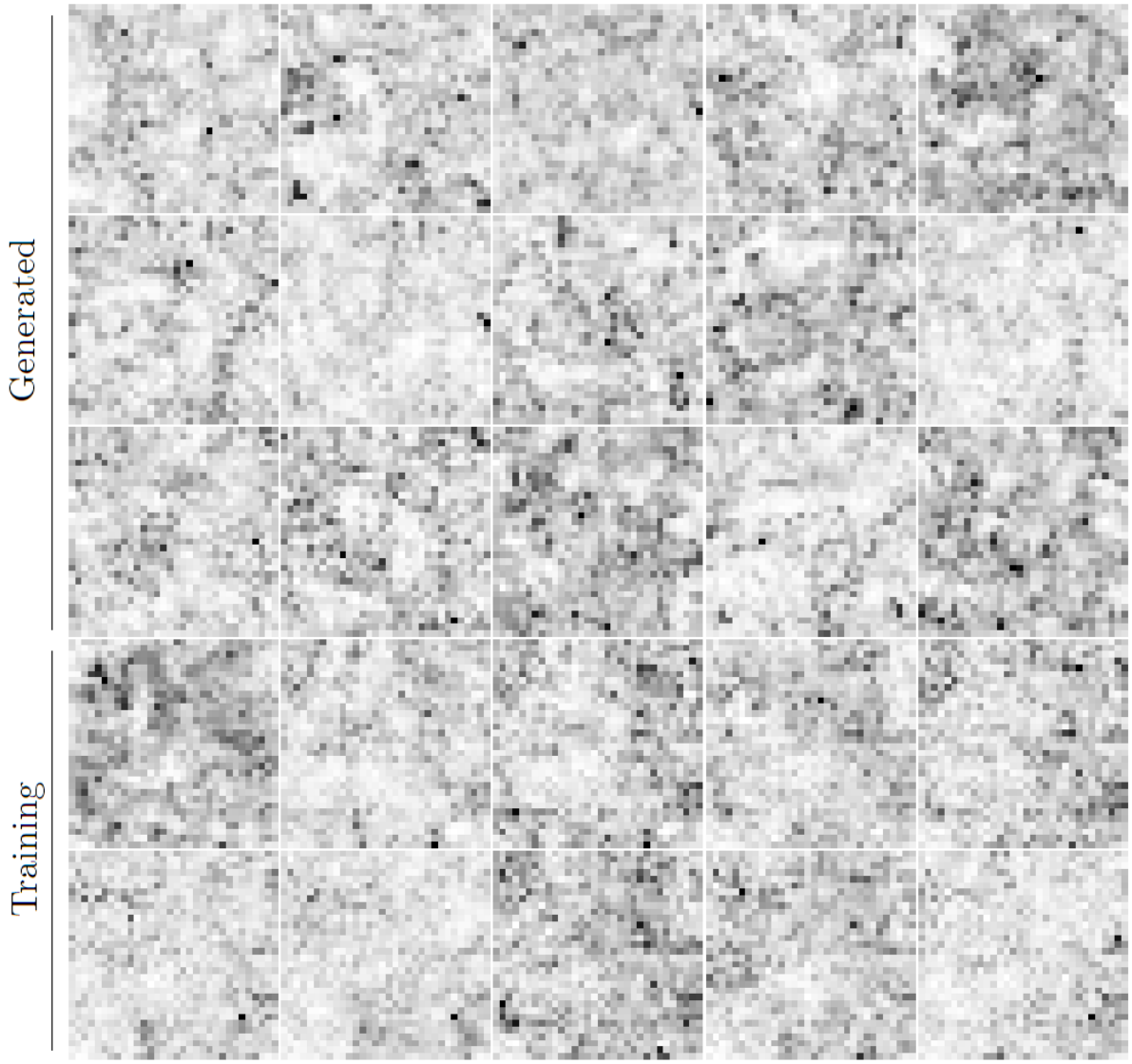

Figure 9: Training data histograms and GAN-generated histograms from the 2D-GAN implementation.

tend to dissipate, causing smaller changes to be made to the samples. The KL divergences between the training and generated distributions t and g, denoted by $D_{KL}(t||g)$ and $D_{KL}(g||t)$, follow a path that is related to the behaviour of the models shown in loss and accuracy curve. The divergences both start at a global maximum and tend to a final convergent value. The values sharply drop at the same epoch in training as the losses and accuracies do. The initial D_{KL} values of the comparison are with earlier generated images that are devoid of structure. The final value, with some intermediate values, corresponds to images that are very similar to the training data, so the D_{KL} values changed over a range of $10^{-1.0}$ to $10^{-2.5}$ from structureless images to realistic images.

8.2.2 3D-GAN implementation

Figure 14 shows the 3D-GAN objectives and accuracies over a run of training. Both the accuracies and the losses each have similar values for all epochs and this is the behaviour of two well matched models. The accuracies of both models converge at just over 60% which implies there is space to improve the model architectures. The separate networks are not able to fully attain their objectives against each other. At this point the discriminator is extracting the maximal amount of information for use in classification and feedback to the generator, which was observed to be changing the histograms only at the scales given by its convolutional kernels. In other words, the generator is fooling the discriminator without generating the most realistic cubes it could. Through running many different models it was found that 8.0×10^4 epochs was enough time for the network to converge. Past this point the training

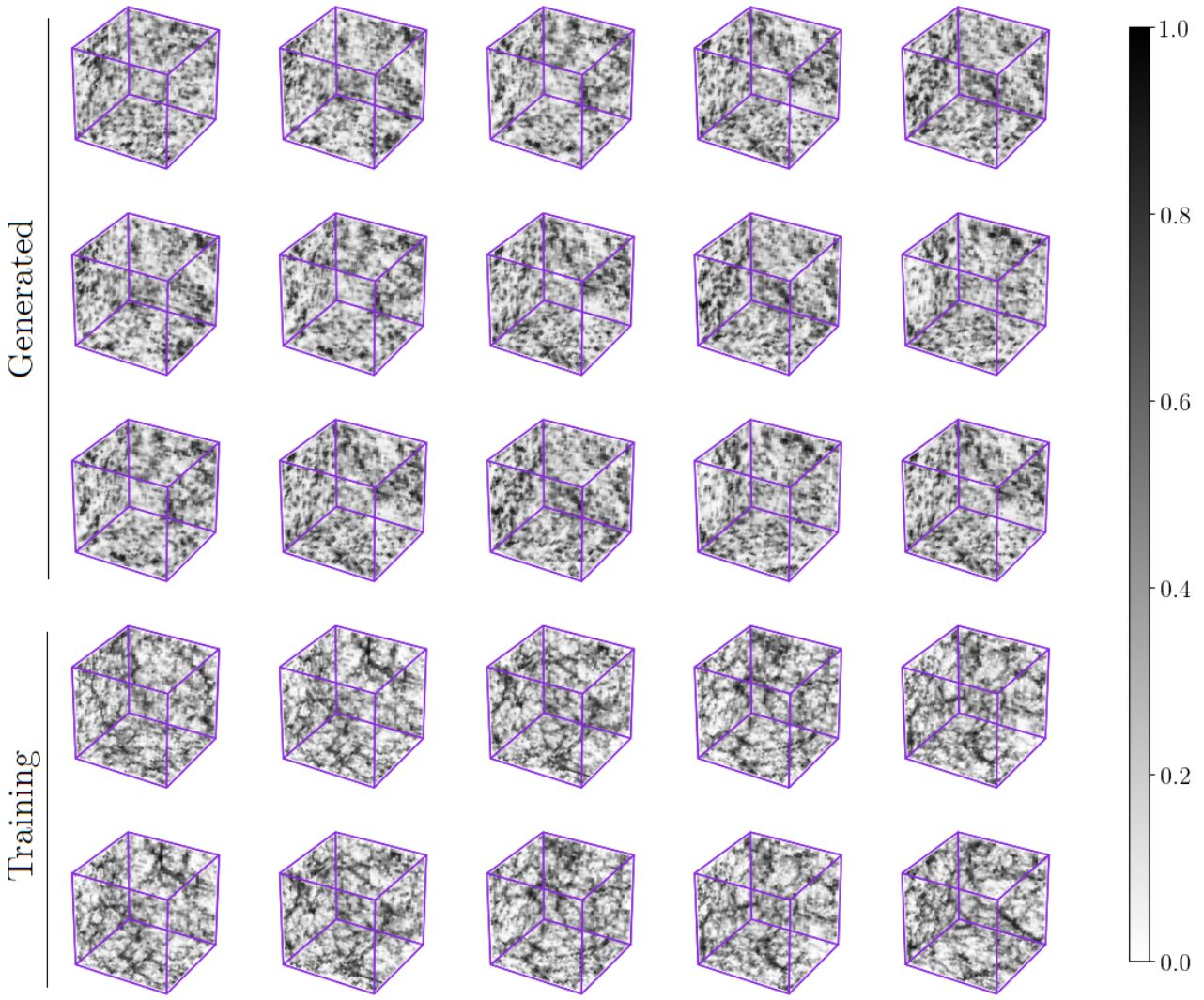

Figure 10: Training data histograms and GAN-generated histograms from the 3D-GAN implementation.

would dissipate chaotically.

Figure 15 shows the KL divergence D_{KL} and KS p-value p_{KS} over a run of training. The curves have similar features to the loss and accuracy curves as they should because of the relation of the cross-entropy to the KL divergence D_{KL}. The initial decrease of both D_{KL} values shows the generator and discriminator learning and the spike at around 4.0×10^4 epochs indicates the generator changing its strategy. The same decrease is seen with a return to similar values of D_{KL}. The KS p-value p_{KS} shows a large change in value over training, particularly when compared to the 2D-GAN p_{KS} curve. Even with the realistic samples of the 2D implementation the null hypothesis that either ensembles of samples originated from the same distribution was always rejected at any point in training. All the p_{KS} values indicated rejection of the null hypothesis at 10% alpha (the significance level of the test, see Appendix D) because the p_{KS} values always fell below the 20% threshold. The small differences between the observed $D_{KL}(t||g)$ and $D_{KL}(g||t)$ values with respect to the values throughout training is important. This suggests that the training and generated distributions t and g are suited for replacement with each other, as opposed to a preferred replacement of one with the other. The fluctuations are larger in the 2D case. Further work is required to comment upon the relative values of the convergent D_{KL} pairs in either implementation. The 3D D_{KL} values changed from the comparisons with initial noisy images against training data from $10^{0.0}$ to $10^{-1.6}$ with the final realistic histograms.

8.3 KL trials

8.3.1 2D-GAN implementation

Figure 16 shows the results of the D_{KL} tests for the 2D-GAN implementation with the D_{KL} values on each plot of the distribution comparisons. There is a consis-

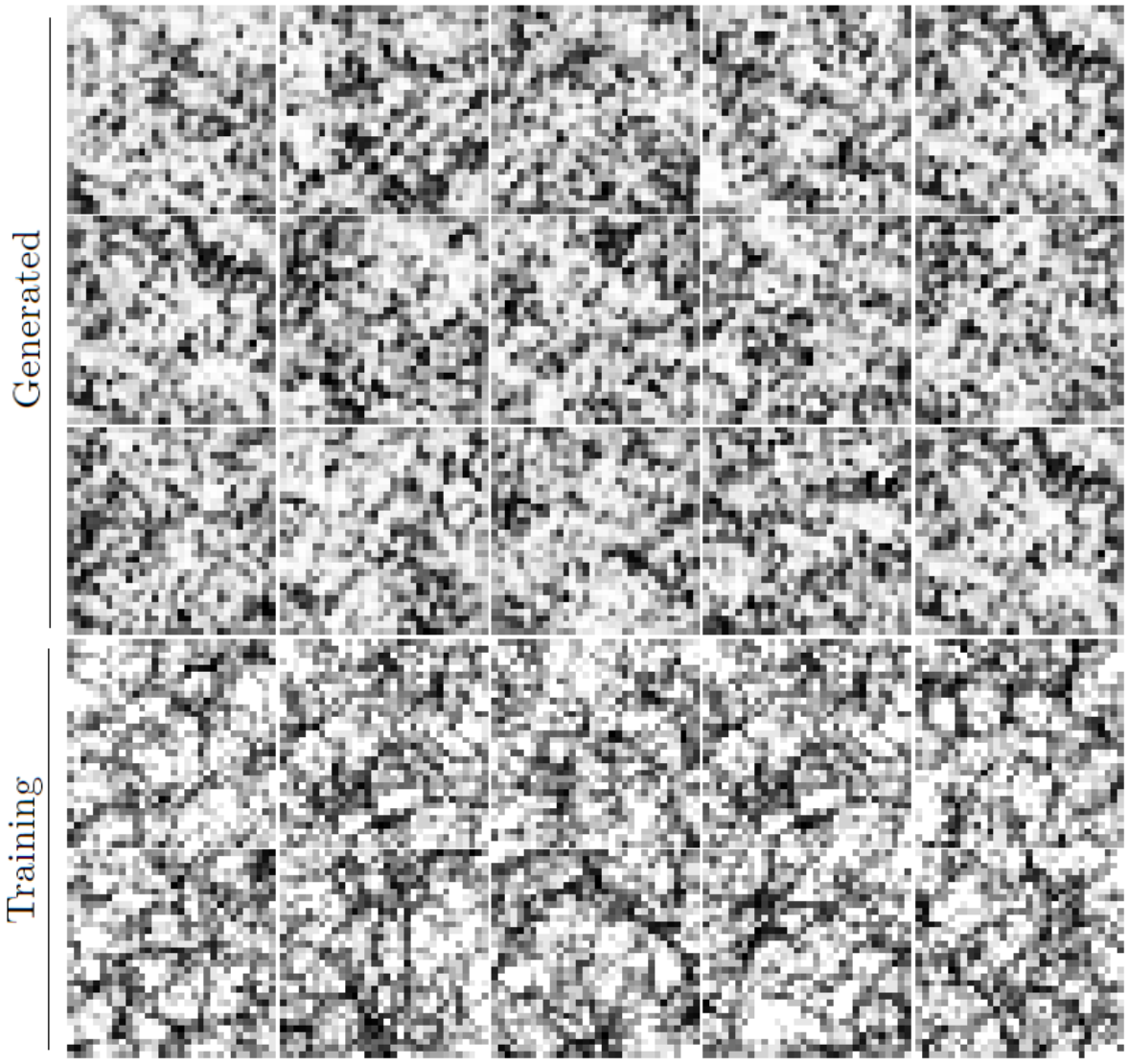

Figure 11: Two-dimensional projections of the 3D-GAN generated histograms.

tent broadening of the histograms along the columns of increasing kernel size. This shows the random effect, or dispersion, on the D_{KL} value from repeated tests to on separate ensembles of histograms. This is caused by the loss of information due to the down-sampling of larger areas which causes more variance in a given value of pixel density because of the averaging over the original pixels. This is predicted by the central-limit theorem. In this discussion a 'mutual comparison' refers to the KL divergence D_{KL} between two different ensembles from the same distribution of t, g or u. The mutual D_{KL} comparisons that show the broadening the most are the generated-generated and uniform-uniform comparisons which together have increasing variance and separation with kernel size. This shows that in both cases the distributions are composed very differently to the uniform distributions. The means of the mutual distributions stay the same except for the uniform-uniform comparison. This is because the median for the bin separations in the component-histograms was given by the median of the training data which has a lower mean pixel density than the uniform histogram data.

The separate kernels cause averaging over different sized areas in the 2D case (volumes in the 3D case). By construction the histograms are already sampling a volume at a fixed resolution for every distribution and so the kernel sizes give the same volume at different levels of resolution. A histogram from any distribution with each kernel down-sampling will exhibit different scales of structure from the original unsampled histogram and this is shown in Figure 7. The ensembles drawn from each distribution are necessary to measure the difference in the structure from the distributions that is not inherent in the individual variance of the images themselves. The latter difference does not concern the distributions of the samples and is less important for the test of the generator capability.

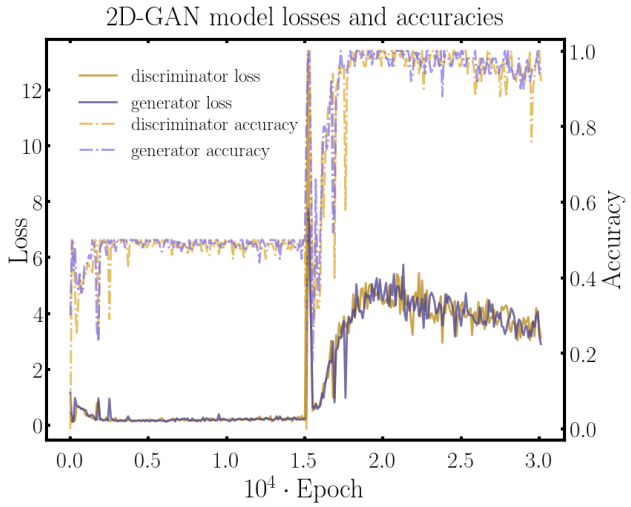

Figure 12: The loss and accuracies curve for the training of the 2D-GAN implementation.

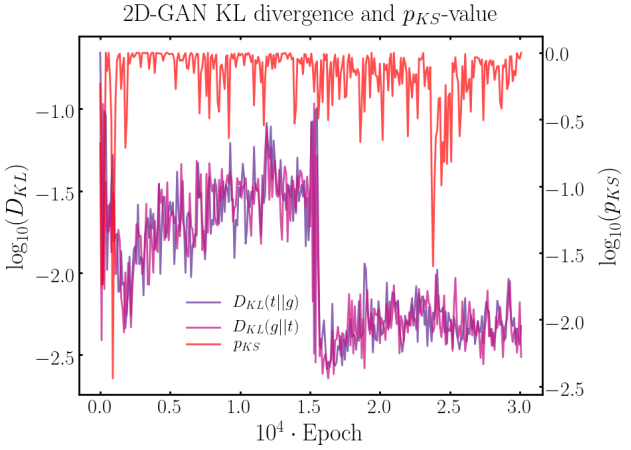

Figure 13: The training against generated KL-divergences and KS p-value curves for the training in the 2D-GAN implementation.

The widths of the mutual comparison histograms in the 2D and 3D D_{KL} trials show that there is an inherent difference in the information content of the ensembles from distributions in the 2D and 3D distributions. This comes from the differences between the samples themselves and with the fixed specifications of the histograms it would be expected that the mutual comparisons show very close values. An important characteristic of the training-generated comparisons is that the D_{KL}-histograms of these distribution pairs overlap. This gives evidence that the distributions can represent each other. The mean $D_{KL}(t||g)$ and $D_{KL}(g||t)$ values are all nearly a whole order of magnitude smaller than the training-uniform and generated-uniform comparison means. These values of the training-uniform and generated-uniform together show that the generator representations are close to the training data because of their similarity in the mean, variance and mean-separations for all values of k. This shows the generator is able to create samples that are strongly correlated to the training data samples.

8.3.2 3D-GAN implementation

The main difference between the 2D and 3D images is in the total information content due to the number of pixels in the 2D and 3D samples. This was a difficult task for the models in the 3D-GAN and the corresponding D_{KL} tests show this. Because of the differences in the total information content a direct comparison between the 2D and 3D D_{KL} trials cannot be made. Given a 2D-GAN and a 3D-GAN both creating realistic samples, it would be expected that the average D_{KL} trials would centre around lower means for the 2D implementation. This is the opposite to what is shown even though the changes in the $D_{KL}(t||g)$ and $D_{KL}(g||t)$ values over training shown in Figures 13 and 15 decrease by the nearly the same amount but for lower starting values in the 2D case.

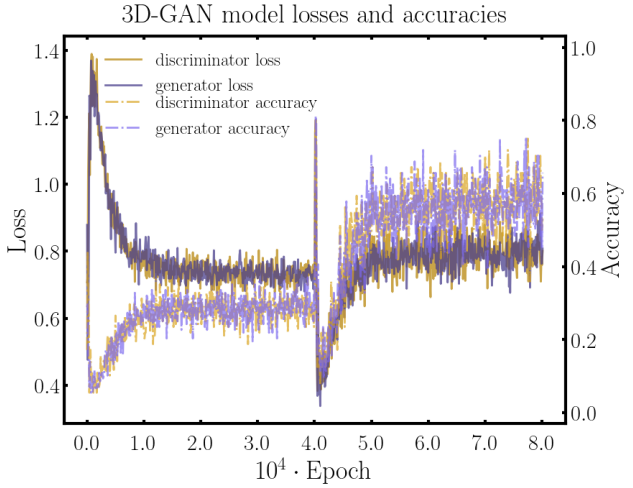

Figure 14: The loss and accuracies curve for the training in the 3D-GAN implementation.

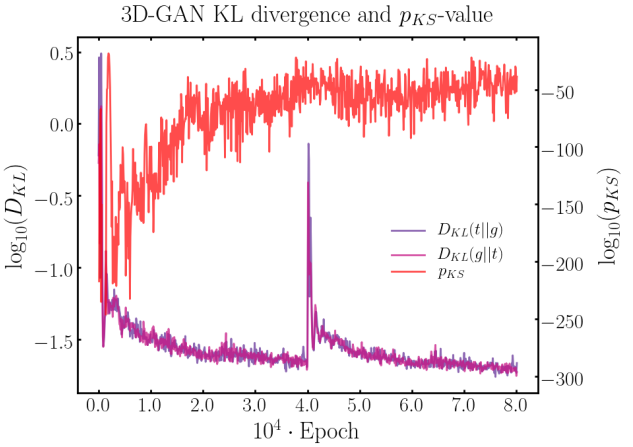

Figure 15: The loss and accuracies curve for the training in the 3D-GAN implementation.

The D_{KL} values of the training-generated comparisons are lower by nearly a whole order of magnitude than the training-uniform and generated-uniform comparisons for all kernel sizes. This is the most important result of this report because it is evidence for the generator being able to generate samples that are statistically similar to the training data samples in the 3D configuration. These values show that much less information would be lost if the model distribution was used in place of the data distribution as opposed to the uniform distribution, the amount lost is around a factor of two more than that of the training distribution approximating itself with different ensembles. This is supported by the horizontal separations of the histogram peaks in each column of the top three rows of Figure 17. The values of the $D_{KL}(t||g)$ and $D_{KL}(g||t)$ are nearly the same as the $D_{KL}(t_1||t_2)$ and $D_{KL}(t_2||t_1)$ values which suggests that replacing the 3D p_{data} distribution with the 3D p_{model} distribution would provide realistic samples for the same potential uses of the training data. This is because the $D_{KL}(t_1||t_2)$ and $D_{KL}(t_2||t_1)$ values are nearly the same as the $D_{KL}(t||g)$ and $D_{KL}(g||t)$ values as well as the fact that they are both smaller than the uniform-generated and uniform-training comparisons.

The mutual D_{KL} comparisons of the training and generated samples show agreement across the different kernels except for a difference of 10^{-3} in the variance of the $k = 4$ comparison. This shows the generators ability to reproduce a variety of structure in the training data samples. The higher D_{KL} average in the $k = 1$ training-generated comparisons are expected; the D_{KL} statistic has more individual pieces of information to compare and find differences with.

The $k = 4$ kernel, whilst being the largest kernel, does not completely measure the continuity of filament structures over a number of the pixels in the $k = 4$ down-sampled histograms. This could be tested of a few larger kernels in histograms with a greater number of total pixels. This could also be measured using a cosmological test such as the two-point correlation function or power spectrum. An increase in the number of bins for the component-histograms would measure the similarity of the generator images to the training data images better. The values of the standard deviation of each D_{KL} histogram relative to the values of the means suggests the number of trials sufficiently sampled the true D_{KL} mean in each test. The empty plot of Figure 17 appears this way, as in the 2D case, because the density bin boundary is lower than the median and mean of the uniform distribution over the pixel densities. The D_{KL} value for this comparison is much lower than 10^{-2}.

The differences between the training-uniform and generated-uniform means and variances are similar to the same values in the 2D-comparisons. These values show the significance of the training-generated comparisons in the top row of Figure 17, which are a nearly

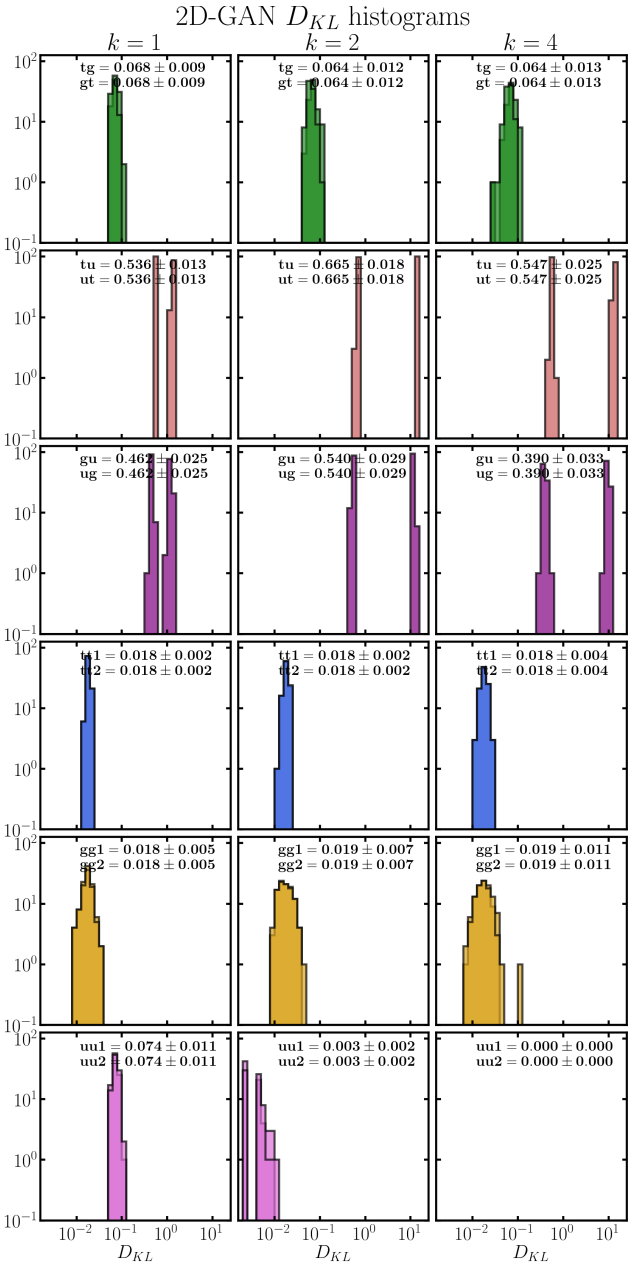

Figure 16: The logarithmic histograms for the KL-trials between the 2D image distributions. Each column holds histograms from a kernel size $k = \{1, 2, 4\}$.

a whole order of magnitude smaller. The mean D_{KL} values of these comparisons are almost at the value of the training-training comparison means which is more evidence for the understandings of the training data by the generator. The similarity shown by the $D_{KL}(t||g)$ and $D_{KL}(g||t)$ values may mostly be due to the similar pixel density distributions of the training and generated data. Considering that the lack of filament structure in the larger scales of the generated data compared to the training data is not shown in the D_{KL} comparisons, this may be true. The pixel density distribution is an important characteristic of the images but the just obtaining the similarity between the t and

g distrubitions with this property would not guarantee that all the structure in the training data was present. There are many combinations of pixels that make up a matching pixel-density distribution for two histogram distributions, but far less of those show the structure of the cosmic web. The higher standard deviations in the D_{KL} histograms for the 2D case in all the kernel sizes compared to the 3D cases is due to the larger size of features relative to total size of the image that shows them in either case.

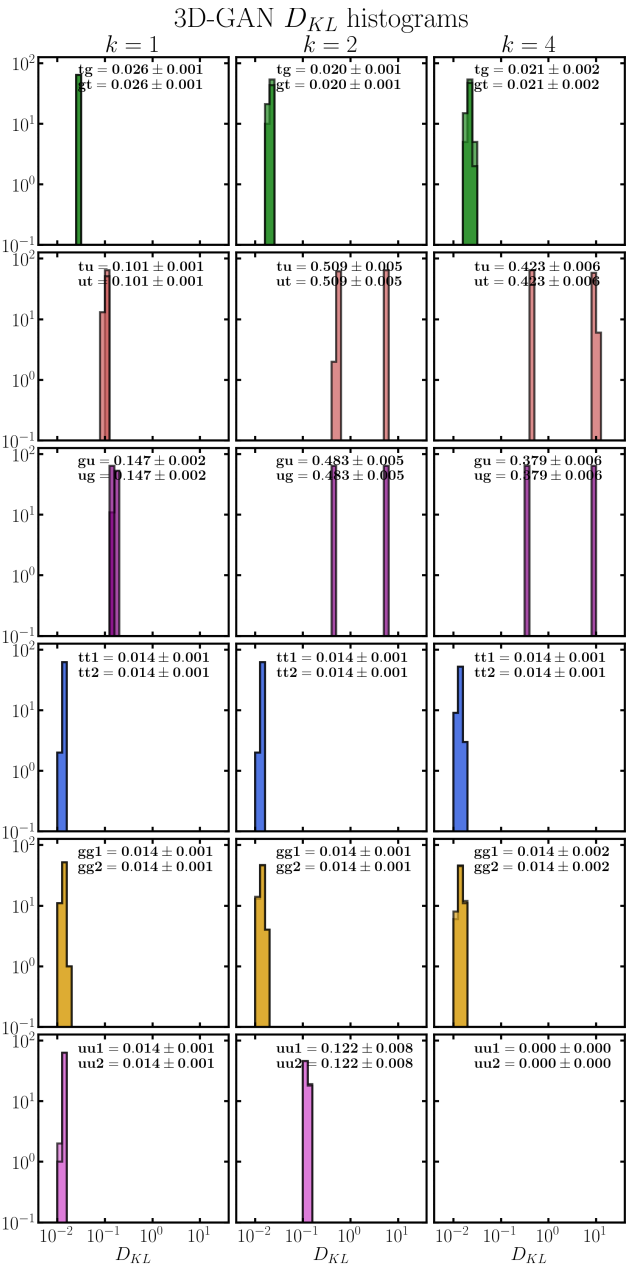

Figure 17: The logarithmic histograms for the KL-trials between the 3D image distributions. Each column holds histograms from a kernel size $k = \{1, 2, 4\}$.

9 Future work

At this outset the capability to generate independent 3D galaxy distributions is the beginning of the full application of generative modelling in teaching a machine to generate an artificial universe. The proceeding sections will show potential research into achieving the full objective.

9.1 Improvements to the present method

The strongest dependence of the quality and realism of the generator product are the GAN models and hyperparameters. It should be noted that each of these should really be tested independently but it is quite possible that a lifetime is not long enough to do so. The author's opinion is that the best route is to improve the current architecture by increasing the number of pixels in the cubes, allowing the depth of the models to increase with more layers, whilst also better translating the variance in the structure inside the histograms. It has been found that in amplifying the initial parameterised units in the first layer of the generator model requires that there are at least $\sim 10^4$ units in the 3D case. The use of fewer units does not allow the generator to draw enough permutations from the potentials 'proto-cubes' allowed by the initial layer. The kernels in the convolutional layers were found to be most effective at size $5\times5\times5$ in the 3D case and the equivalent for the 2D case. Though it is expected that some combination of higher kernel sizes would better characterise the histogram structures.

A larger original simulation with which to sample training data would help the networks process differences between individual samples. The larger volume would give more homogeneous samples with less variance in the summary statistics per sample. The means, total densities and variances of the pixel distributions in each cube were calculated in each iteration of training. Monitoring these statistics showed that the their variance was up to 10% of their total values each. Increasing the homogeneity of the samples would decrease this value. The sizes of the samples relative to the original simulation volume could also be increased from the tested value to increase the homogeneity of the samples. Increasing the sample cube size would also allow the D_{KL} testing to consider more varied structure in the kernel sizes.

The D_{KL} trials could be repeated to compare the 3D two-dimensional cube projections with the 2D generated images and training data. This would connect the results of the separate 2D and 3D implementations. For both of the implementations the p_{KS} testing could have been repeated in the same way as the D_{KL} tests to give more understanding of the differences between the distributions. In this work the p_{KS}-values would be better understood if they were compared to other

distribution comparison p_{KS}-values.

The other available galaxy properties from the Millennium-SAGE catalogs could also be folded into the generator analysis to create distributions weighted by luminosity or another characteristic.

9.2 Conditional Generative Adversarial Networks

The ability to generate different classes of an object does not require separate and independent GANs. The GAN framework may be supplemented by an auxiliary classifier that is attached to the discriminator model, that still classifies the samples as real and fake, whilst additionally characterising the type of the object. This augmented GAN was derived by Odena et al. 2017 [99] and is referred to as an Auxiliary Classifier Generative Adversarial Network (ACGAN). The application to an artificial universe is that the different redshift data available from N-body simulations could be given to an ACGAN. Once this model was trained and had achieved convergence the generator component could create samples at different redshifts. These samples could be tessellated together to form a larger distribution across redshifts. This could follow the environment envisioned by the Millennium Run Observatory (MRObs) [100]. This work sets out to bring together the comparisons of observations and simulations on the same plane of comparison - the perspective of the observer. Figure 18 shows a schematic of how the smaller-scale distributions could be pieced together along the line-of-sight of the observer.

9.3 Redshift and structure interpolation

The latent space from which the generator model samples noise-vectors to change into samples from the model distribution encodes informative features in the natural data samples. This latent space can be navigated, typically along the separate axes, to interpolate through the features. This could be implemented in the latent space of an ACGAN that is trained on data with different redshifts to interpolate between different values of redshift in an inexpensive procedure. This interpolation may hold the ability to evolve a collection of samples simultaneously by interpolating through redshift values in a continuous way. The problem of mode collapse for the generator model could be estimated by interpolating the latent space to create a sample of images to test with each other for similarity. A simple test would be to create the mean-image of this sample to test for overlaps but more advanced cross-correlations could show more detail of the overlaps.

9.4 Universe generation

To create a cosmological volume on order of the size of the Millennium-SAGE volume, with the tested default cube-size, a generator would need to create around 10^2 cubes. The generator model in this work once trained can do so within 30 seconds. Whilst in the process of fusing the samples together coherently avoiding density discontinuities would not be a trivial task. It is logical to assume that it is a process that is maximally as intensive as generating the cubes themselves. It is expected that the GAN would be able to generalise well to higher resolution or larger volume sample data.

The scaled volumes produced by the GAN require the value of the total number of galaxies contained in the cubes to transform the cubes back into coordinate tensors. This would be done by uniformly distributing the number of galaxy positions required for each histogram cell into each cell. The value of the total number of galaxies can be passed to the GAN framework and detached to a separate 'route' in the models. This value, like the histogram cells, would be correctly posited just as the histograms are in training.

9.5 Generative variants

Some of the other possible architectures and variants of the GAN framework could be implemented to see if there was a efficient or more successful model for the objective in this report. A Wasserstein Generative Adversarial Network (WGAN) [44] was tested but it was not found to work as well as the original GAN. The Wasserstein loss function cannot be extended to a categorical version for use in a conditional GAN so if different redshift samples are required for an application the WGAN method would not work anyway.

9.6 Parallelised GANs

One of the problems of training GANs is being sure that the most profitable convergence, best performing architectures and greatest accuracies are obtained. If further work is able to prove that the use of generative models can mimic cosmological structure with enough variety and realism, then the High Performance Computing (HPC) resources used for simulations could be used for creating even larger and more detailed cosmological simulations. The concept has been shown to work for smaller datasets and generative models in [101, 102]. This is particularly important for the case of cosmological simulations because previous simulations involve using assumptions such as the Zel'dovich Approximations [103] and complex numerical codes [61, 104]. Deep learning models are not limited to using low order representations for structural phenomena such as clustering [105]. The datasets available to parallelised generative models can also be far larger and with HPC resources they can use deeper models to process structure at higher resolution.

9.7 Simulation transfer

It is expected that the GAN could perform well on another simulation. The simulations in this work originate from the same ΛCDM cosmology and the webbed structure is consistent. An informative transfer would be to a future simulation operating at a higher resolution with a greater number of total simulation particles such as the Millennium-XXL simulation [26]. The Millennium-XXL simulation uses 6720^3 particles in a volume of $3000\,h^{-1}$ Mpc. Assuming a larger set of samples from this simulation, the GAN would be recreating the distributions of smaller scale cosmological structure that could be tested further. From the experience of obtaining the strongest model in the 3D-GAN implementations, increasing the number of pixels increased the performance of the models. This could be due to a lower loss of information between an increase in the number of pixels for a pixel in the previous resolution. The apperance of the structure is better defined in this way. The lower the number of pixels, the more randomly distributed the pixel densities will be when they average over similar features such as filaments. This is the most difficult obstacle for the GAN as this problem is obviously much greater with the 3D samples. The analog to this problem for images would be the randomness of a set of pixelated images of flowers compared to the same images but in higher resolution. The structure is consistent between the samples in the latter distribution.

With the description above it would also be informative to test the GAN models on a far larger simulation volume with larger cube sizes. The structure is better exemplified if the spectrum of density values is more continuous because larger samples tend to less biased sampling. The improved sampling implies a higher statistical significance of the separate examples of structure in the training data. This is the product of a much more advanced simulation. Different cosmological structure could be definitively shown, some of which could have unexplored origins.

9.7.1 Power Spectrum

The power spectrum P_k is described in Appendix A. This provides a new way of encoding the input simulation data to GAN. It is possible that the GAN could use a procedure outside of training to populate a volume with galaxies if it is given the power spectrum as input data. The three-dimensional fast Fourier transform that would need to be calculated for this task could be done with the `Nbodykit` [106] in future work.

9.8 Statistical diagnostics

The next stage for confirming that the generated data could take the place of the simulation data would be to test the generated data as though it were natural data once the model and data distributions were similar enough. At this point it would be possible for the generated data to offer new insights. The cross-correlations of individual images as well as the derivation of the power-spectra are two methods of further testing the generated data.

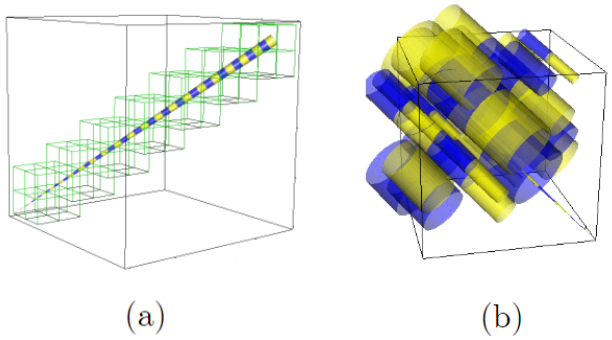

Figure 18: A diagram depicting the pencil beams of an observer outward from their location; (a) - the pencil beam and the required volumes (green) to span the beam with unrepeated galaxy samples, (b) - the highest efficiency for obtaining the volumes of a beam from a distribution volume. From Overzier et al. 2012 [100].

An important task for this method is to estimate the potential for replacing simulation data with generated data. The generator is a fit to the simulation data and sampling a fit does not generate statistically independent samples. In the application of generating a large volume of generated samples it is possible this would not be an issue if the factor of the increase in the total generated volume was not too large.

10 Conclusions

This work has demonstrated the potential of Generative Adversarial Networks in reproducing large-scale cosmological structure with deep convolutional neural networks. The generator model can create realistic samples and this has been achieved in both implementations, but it is still early in terms of stating the generative models as being perfect. In particular this is due to the lack of consistent filament structures in the 3D images. The 2D and 3D implementations have been referenced to each other in rational comparisons, with the generator showing it can imitate simulated cosmological structure. This realism is not limited to a single example, it is present across a large sample of histograms tested over many ensembles from this distribution. This assertion is founded on the statistical tests of Section 8.3 on generated data and simulation-data samples. These tests show that rigorous statistical methods native to natural science are a potential step forward in profiling the behaviour and function of the generative adversarial framework and the field of artificial intelligence at large.

Future work has been outlined that builds on the

foundations of the reported work, moving towards generating an artificial universe at higher levels of statistical agreement with both observation and simulation. The methods of deep learning and generative adversarial networks have been shown to be a valuable tool to extend the expanses of N-body simulations and it seems likely their power will be utilised in the next few years of cosmology.

References

[1] J. R. Bond and et al., "How filaments of galaxies are woven into the cosmic web," *Nature*, vol. 380, p. 603–606, Apr 1996.

[2] P. Coles and L.-Y. Chiang, "Characterizing the nonlinear growth of large-scale structure in the universe," *Nature*, vol. 406, p. 376–378, Jul 2000.

[3] J. E. Forero-Romero and et al., "A dynamical classification of the cosmic web," *Monthly Notices of the Royal Astronomical Society*, vol. 396, p. 1815–1824, Jul 2009.

[4] J. P. Dietrich and et al., "A filament of dark matter between two clusters of galaxies," *Nature*, vol. 487, p. 202–204, Jul 2012.

[5] M. A. Aragón-Calvo and et al., "Multiscale phenomenology of the cosmic web," *Monthly Notices of the Royal Astronomical Society*, vol. 408, p. 2163–2187, Aug 2010.

[6] A. H. Guth, "The Inflationary Universe: A Possible Solution to the Horizon and Flatness Problems," *Phys. Rev.*, vol. D23, pp. 347–356, 1981. [Adv. Ser. Astrophys. Cosmol.3,139(1987)].

[7] D. G. York and et al., "The sloan digital sky survey: Technical summary," *The Astronomical Journal*, vol. 120, p. 1579–1587, Sep 2000.

[8] M. Colless and et al., "The 2df galaxy redshift survey: spectra and redshifts," *Monthly Notices of the Royal Astronomical Society*, vol. 328, p. 1039–1063, Dec 2001.

[9] P. Schneider, "Weak gravitational lensing," *Gravitational Lensing: Strong, Weak and Micro*, p. 269–451, 2006.

[10] M. Kilbinger, "Cosmology with cosmic shear observations: a review," *Reports on Progress in Physics*, vol. 78, p. 086901, Jul 2015.

[11] C. Kiefer and D. Polarski, "Why do cosmological perturbations look classical to us?," 2008.

[12] J. Carlson and et al., "Critical look at cosmological perturbation theory techniques," *Physical Review D*, vol. 80, Aug 2009.

[13] A. H. Guth and S.-Y. Pi, "Fluctuations in the new inflationary universe," *Phys. Rev. Lett.*, vol. 49, pp. 1110–1113, Oct 1982.

[14] T. Abbott and et al., "Dark energy survey year 1 results: Cosmological constraints from galaxy clustering and weak lensing," *Physical Review D*, vol. 98, Aug 2018.

[15] H. Hildebrandt and et al., "Kids-450: cosmological parameter constraints from tomographic weak gravitational lensing," *Monthly Notices of the Royal Astronomical Society*, vol. 465, p. 1454–1498, Nov 2016.

[16] S. Joudaki and et al., "Kids-450: testing extensions to the standard cosmological model," *Monthly Notices of the Royal Astronomical Society*, vol. 471, p. 1259–1279, Apr 2017.

[17] A. A. Klypin and et al., "Dark matter halos in the standard cosmological model: Results from the bolshoi simulation," *The Astrophysical Journal*, vol. 740, p. 102, Oct 2011.

[18] A. J. Benson, "Galaxy formation theory," *Physics Reports*, vol. 495, no. 2, pp. 33 – 86, 2010.

[19] D. Spergel and et al., "Wide-field infrarred survey telescope-astrophysics focused telescope assets wfirst-afta 2015 report," 2015.

[20] R. Laureijs and et al., "Euclid definition study report," 2011.

[21] L. D. E. S. Collaboration, "Large synoptic survey telescope: Dark energy science collaboration," 2012.

[22] S. Alam and et al., "The clustering of galaxies in the completed sdss-iii baryon oscillation spectroscopic survey: cosmological analysis of the dr12 galaxy sample," *Monthly Notices of the Royal Astronomical Society*, vol. 470, p. 2617–2652, Mar 2017.

[23] J. E. Bautista and et al., "The sdss-iv extended baryon oscillation spectroscopic survey: Baryon acoustic oscillations at redshift of 0.72 with the dr14 luminous red galaxy sample," *The Astrophysical Journal*, vol. 863, p. 110, Aug 2018.

[24] H. du Mas des Bourboux and et al., "Baryon acoustic oscillations from the complete sdss-iii ly-quasar cross-correlation function at z = 2.4," *Astronomy Astrophysics*, vol. 608, p. A130, Dec 2017.

[25] M. Boylan-Kolchin and et al., "Resolving cosmic structure formation with the millennium-ii simulation," *Monthly Notices of the Royal Astronomical Society*, vol. 398, p. 1150–1164, Sep 2009.

[26] R. E. Angulo and et al., "Scaling relations for galaxy clusters in the millennium-xxl simulation," *Monthly Notices of the Royal Astronomical Society*, vol. 426, p. 2046–2062, Oct 2012.

[27] P. Nurmi and et al., "Groups in the millennium simulation and in sdss dr7," *Monthly Notices of the Royal Astronomical Society*, vol. 436, p. 380–394, Sep 2013.

[28] T. Karras and et al., "A style-based generator architecture for generative adversarial networks," 2018.

[29] S. Ravanbakhsh and et al., "Enabling dark energy science with deep generative models of galaxy images," 2016.

[30] K. Schawinski and et al., "Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit," *Monthly Notices of the Royal Astronomical Society: Letters*, p. slx008, Jan 2017.

[31] A. C. Rodríguez and et al., "Fast cosmic web simulations with generative adversarial networks," *Computational Astrophysics and Cosmology*, vol. 5, Nov 2018.

[32] M. Mustafa and et al., "Cosmogan: creating high-fidelity weak lensing convergence maps using generative adversarial networks," *Computational Astrophysics and Cosmology*, vol. 6, May 2019.

[33] F.-H. Hsu, *Behind deep blue: building the computer that defeated the world chess champion*. Princeton, N.J: Princeton University Press, 2004. OCLC: 727982790.

[34] E. Alpaydin, *Introduction to Machine Learning*. The MIT Press, 2nd ed., 2010.

[35] F. Chollet, *Deep learning with Python*. Manning Publications Co., 2018.

[36] A. Waibel, "Modular construction of time-delay neural networks for speech recognition," *Neural computation*, vol. 1, no. 1, pp. 39–46, 1989.

[37] G. Tesauro, "Temporal difference learning and td-gammon," *Commun. ACM*, vol. 38, p. 58–68, Mar. 1995.

[38] T. M. Mitchell, *Machine Learning*. USA: McGraw-Hill, Inc., 1 ed., 1997.

[39] I. Goodfellow and et al., *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[40] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. USA: Prentice Hall Press, 3rd ed., 2009.

[41] Ž. Ivezić and et al., *Statistics, Data Mining, and Machine Learning in Astronomy*. Princeton Series in Modern Observational Astronomy, Princeton University Press, 2014.

[42] D. E. Rumelhart and et al., "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[43] I. J. Goodfellow and et al., "Generative adversarial networks," 2014.

[44] M. Arjovsky and et al., "Wasserstein gan," 2017.

[45] T. Karras and et al., "Progressive growing of gans for improved quality, stability, and variation," 2017.

[46] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," 2018.

[47] P. Isola and et al., "Image-to-image translation with conditional adversarial networks," *arxiv*, 2016.

[48] C. Ledig and et al., "Photo-realistic single image super-resolution using a generative adversarial network," 2016.

[49] E. Denton and et al., "Deep generative image models using a laplacian pyramid of adversarial networks," 2015.

[50] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," 2016.

[51] A. Radford and et al., "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015.

[52] Z. Liu and et al., "Deep learning face attributes in the wild," 2014.

[53] F. Yu and et al., "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," 2015.

[54] Krizhevsky and et al., "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[55] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013.

[56] D. J. Rezende and et al., "Stochastic backpropagation and approximate inference in deep generative models," 2014.

[57] C. Ahdida and et al., "Fast simulation of muons produced at the ship experiment using generative adversarial networks," *Journal of Instrumentation*, vol. 14, p. P11028–P11028, Nov 2019.

[58] "Deep generative models for fast shower simulation in ATLAS," Tech. Rep. ATL-SOFT-PUB-2018-001, CERN, Geneva, Jul 2018.

[59] L. de Oliveira, M. Paganini, and B. Nachman, "Learning particle physics by example: Location-aware generative adversarial networks for physics synthesis," *Computing and Software for Big Science*, vol. 1, Sep 2017.

[60] C. Howlett and et al., "L-picola: A parallel code for fast dark matter simulation," 2015.

[61] V. Springel, "The cosmological simulation code gadget-2," *Monthly Notices of the Royal Astronomical Society*, vol. 364, p. 1105–1134, Dec 2005.

[62] J. M. Kratochvil and et al., "Probing cosmology with weak lensing minkowski functionals," *Physical Review D*, vol. 85, May 2012.

[63] X. Yang and et al., "Cosmological information in weak lensing peaks," *Physical Review D*, vol. 84, Aug 2011.

[64] J. M. Kratochvil and et al., "Probing cosmology with weak lensing peak counts," *Physical Review D*, vol. 81, Feb 2010.

[65] V. Springel and et al., "Simulations of the formation, evolution and clustering of galaxies and quasars," *Nature*, vol. 435, p. 629–636, Jun 2005.

[66] Y. Rong and et al., "Massive quiescent galaxies at z$>$ 3 in the millennium simulation populated by a semi-analytic galaxy formation model," *Monthly Notices of the Royal Astronomical Society: Letters*, vol. 471, no. 1, pp. L36–L40, 2017.

[67] P. Bett and et al., "The spin and shape of dark matter haloes in the millennium simulation of a λ cold dark matter universe," *Monthly Notices of the Royal Astronomical Society*, vol. 376, no. 1, pp. 215–232, 2007.

[68] Hilbert, S. and et al., "Ray-tracing through the millennium simulation," *A&A*, vol. 499, no. 1, pp. 31–43, 2009.

[69] L. Wang and et al., "Modelling galaxy clustering in a high-resolution simulation of structure formation," *Monthly Notices of the Royal Astronomical Society*, vol. 371, pp. 537–547, 08 2006.

[70] P. Nurmi and et al., "Groups in the millennium simulation and in sdss dr7," *Monthly Notices of the Royal Astronomical Society*, vol. 436, p. 380–394, Sep 2013.

[71] C. Messenger and J. Read, "Measuring a cosmological distance-redshift relationship using only gravitational wave observations of binary neutron star coalescences," *Phys. Rev. Lett.*, vol. 108, p. 091101, Feb 2012.

[72] A. Palmese and et al., "Gravitational wave cosmology and astrophysics with large spectroscopic galaxy surveys," 2019.

[73] D. Reitze and et al., "Cosmic explorer: The u.s. contribution to gravitational-wave astronomy beyond ligo," 2019.

[74] M. Maggiore and et al., "Science case for the einstein telescope," *Journal of Cosmology and Astroparticle Physics*, vol. 2020, no. 03, p. 050, 2020.

[75] P. Amaro-Seoane and et al., "Laser interferometer space antenna," 2017.

[76] S. Ahmad, R. Myrzakulov, and M. Sami, "Relic gravitational waves from quintessential inflation," *Physical Review D*, vol. 96, Sep 2017.

[77] B. Abbott and et al., "Observation of gravitational waves from a binary black hole merger," *Physical Review Letters*, vol. 116, Feb 2016.

[78] B. P. Abbott and et al., "Gw170817: Observation of gravitational waves from a binary neutron star inspiral," *Phys. Rev. Lett.*, vol. 119, p. 161101, Oct 2017.

[79] B. P. Abbott and et al., "Multi-messenger observations of a binary neutron star merger," *The Astrophysical Journal*, vol. 848, p. L12, oct 2017.

[80] M. Soares-Santos and et al., "The electromagnetic counterpart of the binary neutron star merger LIGO/virgo GW170817. i. discovery of the optical counterpart using the dark energy camera," *The Astrophysical Journal*, vol. 848, p. L16, oct 2017.

[81] C. Guidorzi and et al., "Improved constraints on h 0 from a combined analysis of gravitational-wave and electromagnetic emission from gw170817," *The Astrophysical Journal*, vol. 851, p. L36, Dec 2017.

[82] P. C. N. Aghanim and et al., "Planck 2018 results. vi. cosmological parameters," 2018.

[83] K. P. Mooley and et al., "Superluminal motion of a relativistic jet in the neutron-star merger gw170817," *Nature*, vol. 561, p. 355–359, Sep 2018.

[84] K. Hotokezaka and et al., "A hubble constant measurement from superluminal motion of the jet in gw170817," 2018.

[85] W. L. Freedman and et al., "The carnegie-chicago hubble program. viii. an independent determination of the hubble constant based on the tip of the red giant branch," *The Astrophysical Journal*, vol. 882, p. 34, Aug 2019.

[86] W. Del Pozzo, T. G. F. Li, and C. Messenger, "Cosmological inference using only gravitational wave observations of binary neutron stars," *Phys. Rev. D*, vol. 95, p. 043502, Feb 2017.

[87] R. Gray and et al., "Cosmological inference using gravitational wave standard sirens: A mock data challenge," 2019.

[88] G. Dálya and et al., "Glade: A galaxy catalogue for multimessenger searches in the advanced gravitational-wave detector era," *Monthly Notices of the Royal Astronomical Society*, vol. 479, p. 2374–2381, Jun 2018.

[89] T. L. S. Collaboration, the Virgo Collaboration, the KAGRA Collaboration, and et al., "Prospects for observing and localizing gravitational-wave transients with advanced ligo, advanced virgo and kagra," 2013.

[90] B. Abbott and et al., "Gwtc-1: A gravitational-wave transient catalog of compact binary mergers observed by ligo and virgo during the first and second observing runs," *Physical Review X*, vol. 9, Sep 2019.

[91] D. N. Spergel and et al., "First year wilkinson microwave anisotropy probe observations: Determination of cosmological parameters," *The Astrophysical Journal Supplement Series*, vol. 148, p. 175–194, Sep 2003.

[92] D. J. Croton and et al., "Semi-analytic galaxy evolution (sage): Model calibration and basic results," *The Astrophysical Journal Supplement Series*, vol. 222, p. 22, Feb 2016.

[93] D. J. Croton and et al., "The many lives of active galactic nuclei: cooling flows, black holes and the luminosities and colours of galaxies," *Monthly Notices of the Royal Astronomical Society*, vol. 365, p. 11–28, Jan 2006.

[94] M. Bernyk and et al., "The theoretical astrophysical observatory: Cloud-based mock galaxy catalogs," *The Astrophysical Journal Supplement Series*, vol. 223, p. 9, Mar 2016.

[95] F. Chollet *et al.*, "Keras." https://keras.io, 2015.

[96] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[97] T. Salimans and et al., "Improved techniques for training gans," 2016.

[98] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," 2017.

[99] A. Odena and et al., "Conditional image synthesis with auxiliary classifier gans," 2016.

[100] R. Overzier and et al., "The millennium run observatory: first light," *Monthly Notices of the Royal Astronomical Society*, vol. 428, p. 778–803, Oct 2012.

[101] D. J. Im and et al., "Generative adversarial parallelization," 2016.

[102] S. A. Jacobs and et al., "Parallelizing training of deep generative models on massive scientific datasets," 2019.

[103] Y. B. Zel'Dovich, "Gravitational instability: an approximate theory for large density perturbations.," , vol. 500, pp. 13–18, Mar. 1970.

[104] Y. Feng and et al., "Fastpm: a new scheme for fast simulations of dark matter and haloes," *Monthly Notices of the Royal Astronomical Society*, vol. 463, p. 2273–2286, Aug 2016.

[105] T. Buchert, "Lagrangian theory of gravitational instability - a generic third-order model for non-linear clustering," *Monthly Notices of the Royal Astronomical Society*, vol. 267, p. 811–820, Apr 1994.

[106] N. Hand, Y. Feng, F. Beutler, Y. Li, C. Modi, U. Seljak, and Z. Slepian, "nbodykit: An open-source, massively parallel toolkit for large-scale structure," *The Astronomical Journal*, vol. 156, p. 160, Sep 2018.

[107] D. J. C. MacKay, *Information Theory, Inference Learning Algorithms*. USA: Cambridge University Press, 2002.

[108] P. Virtanen and et al., "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[109] Y. LeCun, "Generalization and network design strategies," 1989.

[110] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

Appendices

A The Power Spectrum P_k

The Cosmological Principle states that the Universe is homogeneous and isotropic. The appearance of the universe in the scales of the images in this report shows that there are voids without luminous matter as well as filaments and clusters of galaxies.

The evolution of the small fluctuations in the early Universe into the webbed volumes of the present day is called structure formation. The exact origins are not well understood and the following argument assumes small perturbations in the energy density of the early universe which were amplified by gravity over cosmic time. These density perturbations obey the wave equation

$$\left(\frac{\partial^2}{\partial t^2} - c_s^2\nabla^2\right)\delta n \tag{A.1}$$

where c_s^2 is the sound speed in the cosmological fluid medium (which at this scale is a fluid regardless of its material composition) and δn is the small change in number density n caused by the perturbation. The complex solutions to the wave equation are

$$\delta n = C(\mathbf{k})\exp(i(\omega t - \mathbf{k}\cdot\mathbf{x})) \tag{A.2}$$

for complex amplitude C, where $\mathbf{k}$ is the co-moving wave vector of the density perturbation in the cosmological fluid, $\mathbf{x}$ is the co-moving vector displacement of the fluid and ω is the angular frequency of this oscillation. A perturbation of this fluid can be described as

$$n(\mathbf{x},t) = \bar{n}(t)[1+\delta(\mathbf{x},t)] \tag{A.3}$$

where $\bar{n}$ is the spatially homogeneous number density and $\delta(\mathbf{x},t)$ is the density contrast given by

$$\delta = \frac{\delta n}{\bar{n}} = \frac{\delta\rho}{\bar{\rho}} \tag{A.4}$$

where ρ and $\bar{\rho}$ are the mass density and its average respectively. The transform to Fourier space is

$$\delta(\mathbf{k},t) = \int d^3x\, \mathrm{e}^{i\mathbf{k}\cdot\mathbf{x}}\,\delta(\mathbf{x},t) \tag{A.5}$$

where $\delta(\mathbf{k})$ is the Fourier transform of $\delta(\mathbf{x})$. The density perturbations in the early Universe can be seen through the spatial distribution of galaxies in the universe. The spatial average of the density contrast $\langle\delta(\mathbf{x},t)\rangle$ is zero at any time t from the isotropy of the universe. The correlation function $\xi(|\mathbf{x}-\mathbf{y}|,t)$ is defined as

$$\xi(|\mathbf{x}-\mathbf{y}|,t) = \langle\delta(\mathbf{x},t),\delta(\mathbf{y},t)\rangle \tag{A.6}$$

This function can be interpreted as the probability of another galaxy at position $\mathbf{y}$ being a distance $|\mathbf{x}-\mathbf{y}|$ from a galaxy at $\mathbf{x}$. $\xi(|\mathbf{x}-\mathbf{y}|,t)$ depends only on $|\mathbf{x}-\mathbf{y}|$ since the universe is homogeneous and isotropic. The scalar value of the separation gives the isotropic probability. The correlation function is a measure of the degree of clustering in the spatial distribution of galaxies compared to a random distribution of the galaxies.

Returning to the Fourier transform of the density contrast $\delta(\mathbf{x},t)$, the correlation function in $\mathbf{k}$-space is

$$\langle\delta(\mathbf{k},t),\delta(\mathbf{k}',t)\rangle = 8\pi^3\delta_D^3(\mathbf{k}+\mathbf{k}')\int d^3r\, \mathrm{e}^{i\mathbf{k}\cdot\mathbf{r}}\,\xi(r,t) \tag{A.7}$$

where δ_D^3 is the three-dimensional Dirac-delta function and $\mathbf{r}=\mathbf{x}-\mathbf{y}$. The definition of the power spectrum P_k is

$$P_k = \int d^3r\, \mathrm{e}^{i\mathbf{k}\cdot\mathbf{r}}\,\xi(r,t) \tag{A.8}$$

which shows the power spectrum is the three-dimensional Fourier transform of the correlation function. This quantity in either a two or three-dimensional space, with the density field fluctuations being drawn from a Gaussian distribution, means that the power spectrum gives a distribution of the fluctuations. The power spectrum describes the amplitude of fluctuations on different length scales or equivalently mass scales.

B GAN architectures and hyper-parameters

The model architectures of the 2D- and 3D-GAN models are shown in Tables 1 and 3 respectively. The hyper-parameters of the 2D- and 3D-GAN implementations are shown in Tables 2 and 4 respectively.

	Activation	**Output Shape**
Generator		
z		(100)
Linear	tanh	(1024)
Linear	tanh	(16, 16, 128)
Conv2D	tanh	(16, 16, 64)
Conv2D	tanh	(32, 32, 1)
Discriminator		
x		(32, 32, 1)
Conv2D	tanh	(32, 32, 128)
Conv2D	tanh	(16, 16, 64)
Linear	tanh	(1024)
Linear	sigmoid	1

Table 1: Architectures for the generator and discriminator networks in the 3D-GAN.

Hyper-parameter	**Value**
Discriminator learning rate	2×10^{-4}
Discriminator dropout factor	0.5
Generator learning rate	2×10^{-4}
z dimension	100
Prior bounds	$[-1,+1]$
Batch size / total samples	64 / 128

Table 2: Hyper-parameters used in 2D-GAN implementation.

	Activation	**Output Shape**
Generator		
z		(200)
Linear	BatchNorm Relu	(16, 16, 16, 512)
Conv3D $(5\times5\times5)$	BatchNorm Relu	(16, 16, 16, 512)
Conv3D $(5\times5\times5)$	BatchNorm Relu	(32, 32, 32, 256)
Conv3D $(5\times5\times5)$	BatchNorm tanh	(32, 32, 32, 1)
Discriminator		
x		(32, 32, 32, 1)
Conv3D $(5\times5\times5)$	BatchNorm LeakyRelu	(32, 32, 32, 1)
Conv3D $(5\times5\times5)$	BatchNorm LeakyRelu	(16, 16, 16, 256)
Conv3D $(5\times5\times5)$	BatchNorm LeakyRelu	(8, 8, 8, 512)
Linear	sigmoid	1

Table 3: Architectures for the generator and discriminator networks in the 3D-GAN.

C Kullback-Leibler Divergence

C.1 Information theory

Information theory aims to quantify how much information is in a signal. The basic principle is that learning about an unlikely event is more informative than learning about a likely event - the information content of the unlikely event is higher than the unlikely event [39]. This means that learning about an event that consists of two unlikely events contains more information content than learning of one of the two events

Hyper-parameter	Value
Discriminator learning rate	2×10^{-6}
Discriminator dropout factor	0.3
Generator learning rate	2×10^{-6}
z dimension	200
Prior bounds	$[-1, +1]$
Batch size / total samples	64 / 128

Table 4: Hyper-parameters used in 3D-GAN implementation.

alone. The quantity of information content is expressed by definition of the self-information of an event $\mathrm{x} = x$ as

$$I(x) = -\log P(x) \tag{C.1}$$

where the units of the self-information are depend on the base of the logarithm this equation. The same definition of information for continuous events x applies as for the discrete events.

Self-information is associated with outcomes from singular events. The amount of uncertainty in a probability distribution is given by the Shannon entropy [107]

$$H(x) = \mathbb{E}_{\mathrm{x}\sim P}[I(x)] = -\mathbb{E}_{\mathrm{x}\sim P}[\log P(x)] \tag{C.2}$$

this definition shows that the Shannon entropy is the expectation value of the amount of information in an event drawn from that distribution. When x is continuous the Shannon entropy is called the differential entropy.

C.2 The Kullback-Leibler Divergence

A measure for the difference in the information content of two probability distributions $P(\mathrm{x})$ and $Q(\mathrm{x})$ over the same random variable x is given by the Kullback-Leibler (KL) divergence

$$\begin{aligned} D_{KL}(P||Q) &= \mathbb{E}_{\mathrm{x}\sim P}\left[\log \frac{P(x)}{Q(x)}\right] \\ &= \mathbb{E}_{\mathrm{x}\sim P}[\log P(x) - \log Q(x)] \end{aligned} \tag{C.3}$$

The KL divergence is clearly only positive and only equal to zero if the distributions are exactly the same. For continuous variables the zero value for the case of two identical distributions is for 'almost everywhere' - the relation holds everywhere in space except on a set of measure zero. The KL divergence is not a symmetric quantity; $D_{KL}(P||Q) \neq D_{KL}(Q||P)$.

C.3 Cross-entropy

The sum of the KL divergence for P and Q with the Shannon entropy of P gives the cross-entropy $H(P, Q)$ of P and Q as

$$\begin{aligned} H(P, Q) &= H(P) + D_{KL}(P||Q) \\ &= -\mathbb{E}_{\mathrm{x}\sim P} \log Q(x) \end{aligned} \tag{C.4}$$

Note that minimizing the cross-entropy with respect to Q is equivalent to minimizing the KL divergence because the differential entropy of P is independent of Q [39].

The objective function $J^{(D)}$ defined in Section 3 means the discriminator tries to maximize the probability of correctly classifying real and fake samples. This is the same as maximizing two separate cross-entropies. The first is the cross-entropy of the discriminator classifying the real samples from the data distribution p_{data} as real. The second is the cross-entropy of the discriminator of classifying the fake samples from the generator distribution p_{model} as fake.

D Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov (KS) test can show whether a sample comes from a population with a specific distribution or not. It is based on the empirical distribution function (EDF) which is defined, for N magnitude-ordered data points $\{x_i\}$, as

$$E_N = \frac{n(i)}{N} \tag{D.1}$$

where $n(i)$ is the number of points with values less than x_i. It has been assumed that $\{x_i\}$ are independent and identically distributed; in other words, that the samples $\{x_i\}$ are independent of each other and are all drawn from the same probability distribution. With a model cumulative distribution function (CDF) $R(x)$ to test the cumulative data distribution function against, the KS test statistic D_{KS} is

$$D_{KS} = \max |R(x) - S(x)| \tag{D.2}$$

So D_{KS} is the greatest difference between the model CDF $R(x)$ and the EDF of the data $S(x)$. The model CDF represents the distribution to test for the null hypothesis. For this work this is the comparison of the mean image of the data-generating distribution p_{data} to the mean image of the generated data distribution p_{model}. The null hypothesis is the default statement on the comparison that there is no significant correlation. Figure 19 shows two data distributions and a model distribution with comparisons of the corresponding EDFs and CDF.

The KS test p-value p_{KS} is calculated during GAN training. The p-value is defined as the probability of obtaining test results at least as extreme as the results actually observed during the test, assuming that the null hypothesis is correct. This translates to quoting the probability of obtaining the KS test statistic D_{KS} given that the distributions are not correlated. The

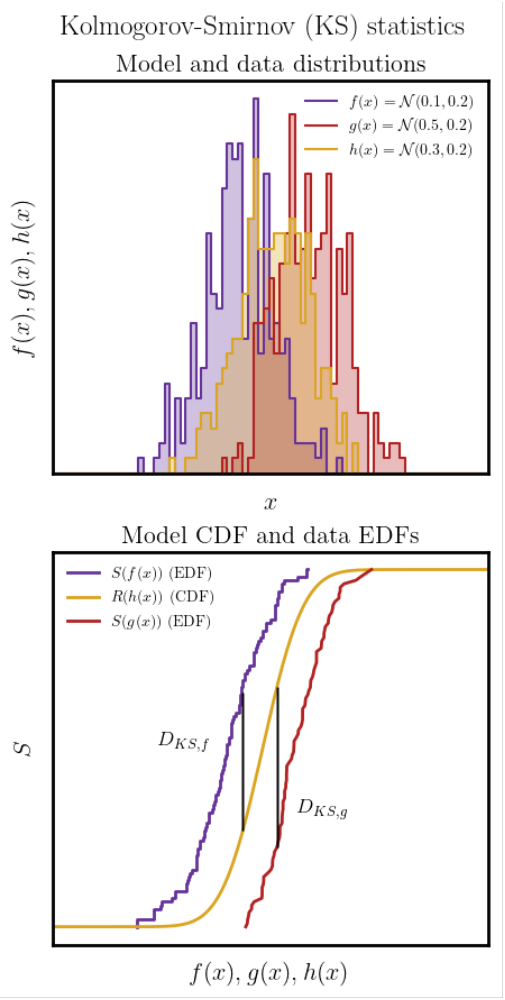

Figure 19: An illustration of the KS statistic for two data distributions $f(x)$ and $g(x)$ and one model distribution $h(x)$. The model cumulative distribution function (CDF) and data empricial distribution functions (EDFs) are shown.

strength of the KS test is that no underlying population distribution function is asserted before calculating p_{KS}. The p_{KS} value is referenced against an α-value that thresholds the value to reject the null hypothesis with. For $\alpha = 0.01$ and a sample size of 64 for each of the compared ensembles in the training loop determinations of p_{KS}, this provides a p-value of 0.20. In this work all the KS p-values were calculated using the SciPy [108] `stats.ks_2samp` function.

E GAN components

E.1 Convolutional layers

The standard unit in the layers of neural networks is the linear unit of general matrix multiplication. Convolutional neural networks (CNNs) pioneered by LeCun [109] were the first design for methods that could decode grid-like data such as images. The procedures in CNNs use a linear convolution operation as opposed to the multiplications in linear units that are also known as perceptrons. The convolution operation between two functions f and g is defined as

$$h(x) = (f * g)(x) = \int f(\tau)g(x - \tau)d\tau \tag{E.1}$$

where for an image the function f in the convolution is known as the input, the function g is the kernel and the output function h is the feature map. The grid data means the convolution is discrete so that

$$h(x) = (f * g)(x) = \sum f(\tau)g(x - \tau) \tag{E.2}$$

In machine learning the input is an array of data and the kernel is an array of parameters that the learning algorithm changes in training. In the 3D-GAN the convolution equation takes the form

$$\begin{aligned} S(i,j,k) &= (I * K)(i,j,k) \\ &= \sum_{l,m,n} I(i+l, j+m, k+n)K(l,m,n) \end{aligned} \tag{E.3}$$

for the 3D case, where I and K are the image and kernel grids and S is the feature map from the convolution. A similar equation holds for the 2D case. The indices l, m and n are the kernel size in each dimension. Because convolutions are commutative, the kernel in the discrete convolutions passes over the image (the kernel and image have been commuted), these features can be recorded in a network. Figure 20 shows how a feature map is taken from an image.

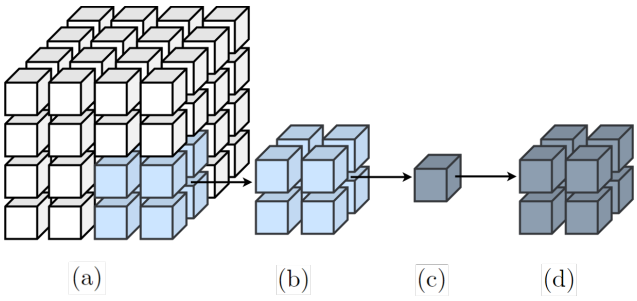

Figure 20: A general 3D-convolution operation; (a) - the image inputted to the convolutional layer with the kernel at one position in the image, (b) - the kernel-image convolution output at a position, (c) - the output pixel from the operation and (d) - the feature map of convolutions across the image.

E.2 Activation functions

The activation functions used in each layer of the models in the generator and discriminator are important

for the performance of the GAN. Figure 21 shows the activation functions used in the models, plotted on $x, y \in [-1, +1]$.

In the generator models rectified linear unit (ReLU) activation functions were put at the end of the inner layers with a hyperbolic tangent activation (tanh) activation at the end of the last layer. This final activation helps the GAN match the pixel counts to the range of the training data samples. LeakyReLU activations are the same as ReLU functions for positive inputs but are scaled by a value α for negative inputs. The value of $\alpha = 0.2$ was taken from [51]. On the end of the discriminator network a sigmoid activation function was used to match the range of probabilities $P \in [0, 1]$.

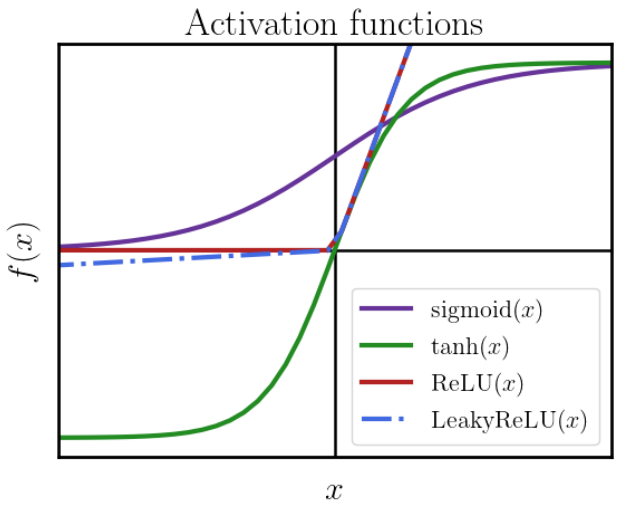

Figure 21: The four types of activation function used in the GAN models.

E.3 Batch-normalisation

The forward flow of information in a deep learning model means the parameters of the previous layer influence the parameters of the next layer through changing the inputs to the next layer. The gradients in optimisation are used to update the whole model simultaneously which can have unexpected effects because the updates are calculated under the assumption that the other layers are held constant. Ioffe and Szegedy 2015 [110] proposed a reparameterization to help apply the updates across separate layers. Batch normalisation involves normalising data in the batches that it is presented to a layer in [39]. For a batch of layer activations expressed as a matrix $\mathbf{H}$, normalising requires replacement of $\mathbf{H}$ by

$$\mathbf{H}' = \frac{\mathbf{H} - \mu}{\sigma} \tag{E.4}$$

where μ is a vector containing the mean of each activation and σ is a vector containing the standard deviation of each activation. Each row in $\mathbf{H}$ corresponds to the activations of some layer for every example in the batch. This means Equation E.4 applies σ and μ to each row in $\mathbf{H}$. The network operates on $\mathbf{H}'$ just as it did on $\mathbf{H}$. In the training loop

$$\begin{aligned} \mu &= \frac{1}{m}\sum_i \mathbf{H}_i \\ \sigma &= \sqrt{\delta + \frac{1}{m}\sum_i (\mathbf{H} - \mu)_i^2} \end{aligned} \tag{E.5}$$

where δ is a constant. These equations give the same parameters in Equation E.4 for a normalised and reparameterized $\mathbf{H}$.

E.3.1 Stochastic Gradient Descent

A gradient descent algorithm outlines the process of minimizing an objective function for a model. The function may be distributive over a set of samples so that it is the same as the sum of the function acting on each sample. Stochastic gradient descent (SGD) is a common optimisation algorithm. It uses a constant learning rate (a step to move through the parameter space with) per iteration of training, though in practice this value depends on the value of the iteration for more advanced algorithms.

In SGD a sample of m latent-space vectors $\{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ are drawn from $p_\mathbf{z}$ with a sample of m samples of real data $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$. The discriminator parameters are updated the ascension of the stochastic gradient

$$\nabla_{\theta^{(D)}} \frac{\epsilon}{m} \sum_{i=1}^{m} \left[\log D(\mathbf{x}^{(i)}) + \log(1 - D(G(\mathbf{z}^{(i)}))) \right] \tag{E.6}$$

and generator is updated by the descension of its stochastic gradient

$$\nabla_{\theta^{(G)}} \frac{\epsilon}{m} \sum_{i=1}^{m} \log(1 - D(G(\mathbf{z}^{(i)}))) \tag{E.7}$$

where ϵ is the learning rate and $\nabla_{\boldsymbol{\theta}}$ is the directional derivative in the direction of increasing or decreasing gradient.

Acknowledgements

Thank you to Prof. Martin Hendry and Dr. Chris Messenger for their time and guidance. I learnt a lot that I will use in the future. I am inspired and I look forward to researching AI in cosmology in the future. Thank you to Michael Williams and Jordan 'McGAN' McGinn for their insights and discussions.

Data and code availability

All code can be found at https://github.com/JedHmr/universe_gen. Some extra material can be found on http://www.astro.gla.ac.uk/users/jed/