The Astrophysical Journal Supplement Series, 259:61 (14pp), 2022 April https://doi.org/10.3847/1538-4365/ac5ab0

The CAMELS Multifield Data Set: Learning the Universe's Fundamental Parameters with Artificial Intelligence

Francisco Villaescusa-Navarro[1,2], Shy Genel[2,3], Daniel Anglés-Alcázar[2,4], Leander Thiele[5], Romeel Dave[6,7,8], Desika Narayanan[9,10], Andrina Nicola[1], Yin Li[2,11], Pablo Villanueva-Domingo[12], Benjamin Wandelt[2,13], David N. Spergel[1,2], Rachel S. Somerville[2], Jose Manuel Zorrilla Matilla[1], Faizan G. Mohammad[14,15], Sultan Hassan[2,7], Helen Shao[1], Digvijay Wadekar[16,17], Michael Eickenberg[11], Kaze W. K. Wong[2], Gabriella Contardo[2], Yongseok Jo[18], Emily Moser[19], Erwin T. Lau[20], Luis Fernando Machado Poletti Valle[21], Lucia A. Perez[22], Daisuke Nagai[21,23], Nicholas Battaglia[19], and Mark Vogelsberger[24]

[1] Department of Astrophysical Sciences, Princeton University, Peyton Hall, Princeton, NJ 08544, USA; villaescusa.francisco@gmail.com
[2] Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, New York, NY 10010, USA
[3] Columbia Astrophysics Laboratory, Columbia University, New York, NY 10027, USA
[4] Department of Physics, University of Connecticut, 196 Auditorium Road, Storrs, CT 06269, USA
[5] Department of Physics, Princeton University, Princeton, NJ 08544, USA
[6] Institute for Astronomy, University of Edinburgh, Royal Observatory, Edinburgh EH9 3HJ, UK
[7] Department of Physics & Astronomy, University of the Western Cape, Cape Town 7535, South Africa
[8] South African Astronomical Observatories, Observatory, Cape Town 7925, South Africa
[9] Department of Astronomy, University of Florida, Gainesville, FL, USA
[10] University of Florida Informatics Institute, 432 Newell Drive, CISE Bldg E251, Gainesville, FL, USA
[11] Center for Computational Mathematics, Flatiron Institute, 162 5th Avenue, New York, NY 10010, USA
[12] Instituto de Física Corpuscular (IFIC), CSIC-Universitat de València, E-46980, Paterna, Spain
[13] Sorbonne Universite, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis boulevard Arago, F-75014 Paris, France
[14] Waterloo Center for Astrophysics, University of Waterloo, Waterloo, ON N2L 3G1, Canada
[15] Department of Physics and Astronomy, University of Waterloo, Waterloo, ON N2L 3G1, Canada
[16] Center for Cosmology and Particle Physics, Department of Physics, New York University, New York, NY 10003, USA
[17] School of Natural Sciences, Institute for Advanced Study, Princeton, NJ 08540, USA
[18] Center for Theoretical Physics, Department of Physics and Astronomy, Seoul National University, Seoul 08826, Republic of Korea
[19] Department of Astronomy, Cornell University, Ithaca, NY 14853, USA
[20] Center for Astrophysics | Harvard & Smithsonian, 60 Garden Street, Cambridge, MA 02138, USA
[21] Department of Physics, Yale University, New Haven, CT 06520, USA
[22] Arizona State University, School of Earth and Space Exploration, 781 Terrace Mall, Tempe, AZ 85287, USA
[23] Department of Astronomy, Yale University, New Haven, CT 06511, USA
[24] Kavli Institute for Astrophysics and Space Research, Department of Physics, MIT, Cambridge, MA 02139, USA
Received 2021 October 4; revised 2022 February 28; accepted 2022 March 2; published 2022 April 7

Abstract

We present the Cosmology and Astrophysics with Machine Learning Simulations (CAMELS) Multifield Data set (CMD), a collection of hundreds of thousands of 2D maps and 3D grids containing many different properties of cosmic gas, dark matter, and stars from more than 2000 distinct simulated universes at several cosmic times. The 2D maps and 3D grids represent cosmic regions that span ∼100 million light-years and have been generated from thousands of state-of-the-art hydrodynamic and gravity-only *N*-body simulations from the CAMELS project. Designed to train machine-learning models, CMD is the largest data set of its kind containing more than 70 TB of data. In this paper we describe CMD in detail and outline a few of its applications. We focus our attention on one such task, parameter inference, formulating the problems we face as a challenge to the community. We release all data and provide further technical details at https://camels-multifield-dataset.readthedocs.io.

Unified Astronomy Thesaurus concepts: Cosmological parameters from large-scale structure (340); Magnetohydrodynamical simulations (1966); Astrostatistics (1882); N-body simulations (1083)

1. Introduction

In the era of precision cosmology, there is great interest in developing sophisticated techniques to optimally measure cosmological parameters from astrophysical data sets. To achieve the desired precision of next-generation experiments often requires accounting for complex astrophysical effects that can impact the properties of the luminous galaxies and gas from which cosmological parameters are inferred.

Advances in deep learning are triggering a revolution in many different disciplines, from biology to social sciences. In cosmology, deep learning is being used to carry out many different complex tasks where traditional methods were slow, inaccurate, or simply nonexistent. Examples of such tasks are speeding up simulations (He et al. 2019; Alves de Oliveira et al. 2020), obtaining superresolution simulations (Kodi Ramanah et al. 2020; Li et al. 2021; Ni et al. 2021), cleaning up astrophysics (Makinen et al. 2021; Villanueva-Domingo & Villaescusa-Navarro 2021), painting stars and gas properties on the dark matter field (Modi et al. 2018; Jo & Kim 2019; Tröster et al. 2019; Yip et al. 2019; Zhang et al. 2019; Kasmanoff et al. 2020; Thiele et al. 2020; Harrington et al. 2021; Moews et al. 2021; Wadekar et al. 2021), changing the cosmology of a simulation (Giusarma et al. 2019), generating new data with desired properties (Böhm & Seljak 2020; Dai & Seljak 2021), predicting halo masses (Villanueva-Domingo et al. 2021a, 2021b), and detecting anomalies (Storey-Fisher et al. 2021), among many more.[25]

One of the main reasons behind this success is the fact that neural networks behave as universal function approximators (Cybenko 1989; Hornik et al. 1990; Hornik 1991). Differently to many traditional methods, neural networks do not require the use of analytic expressions and are not limited to low-dimensional spaces. On the other hand, training neural networks usually requires very large data sets. While existing data sets for computer vision tasks are numerous and diverse (e.g., MNIST,[26] CIFAR10,[27] and ImageNet[28]), the situation is very different for cosmology.

Large-volume cosmological hydrodynamic simulations have become a primary tool to study the formation of galaxies and large-scale structure (Somerville & Davé 2015). These simulations follow explicitly the evolution of the dark matter and baryonic components in an expanding universe, incorporating a variety of subgrid prescriptions to model key physical processes such as star formation and feedback from massive stars and the impact of active galactic nuclei (AGNs) feedback powered by accretion onto supermassive black holes (e.g., Genel et al. 2014; Anglés-Alcázar et al. 2017a; Pillepich et al. 2018a; Davé et al. 2019). However, many of these processes remain poorly understood and require careful tuning of free parameters. While a single simulation can provide a plausible representation of the universe, model-dependent parameter degeneracies limit the predictive power of simulations and their use as primary tool to produce robust constraints on cosmological parameters.

The Cosmology and Astrophysics with Machine Learning Simulations (CAMELS) project (Villaescusa-Navarro et al. 2021a) has pioneered a new approach, producing thousands of cosmological hydrodynamic simulations to train machine-learning algorithms, varying cosmology, initial random field, subgrid galaxy formation model, and stellar and AGN feedback parameters.

In this paper we present and make publicly available the CAMELS Multifield Data set (CMD), a large data set containing hundreds of thousands of 2D maps and 3D grids with different properties obtained from 2000 distinct universes. Each 2D map and 3D grid represents a region with an area of $(25\,h^{-1}\,\mathrm{Mpc})^2$ and volume of $(25\,h^{-1}\,\mathrm{Mpc})^3$, respectively, where many different fields are included, from dark matter to gas and stellar properties, at different redshifts. Each 2D map and 3D grid is associated with a vector of either two or six numbers: two cosmological parameters (all data) and four astrophysical parameters (only data from hydrodynamic simulations) that define and control the behavior of the parent cosmological simulation in each case. The full data set comprises more than 70 TB and represents the largest collection of 2D maps and 3D grids from state-of-the-art hydrodynamic simulations publicly available to date, and may serve as a standard cosmological and astrophysical data set to train machine-learning models to perform a large variety of tasks.

In our companion papers (Villaescusa-Navarro et al. 2021b, 2021c), we used CMD to perform, for the first time, likelihood-free inference of cosmological parameters at the field level from 2D maps generated from state-of-the-art hydrodynamic simulations of 13 different fields, obtaining very promising results. In this work we describe in detail the architecture and training procedure used in those works. Furthermore, we formulate the problems we encountered as a challenge to the community. We also release all codes and network weights from our companion papers (Villaescusa-Navarro et al. 2021b, 2021c) as a benchmark to compare against.

Information on how to download and manipulate CMD, together with the codes and network weights used for parameter inference and other tasks can be found at https://camels-multifield-dataset.readthedocs.io.

This paper is organized as follows. In Section 2 we describe CMD in detail. We then outline in Section 3 a few of its possible applications, focusing our attention on parameter inference. We conclude in Section 4.

2. Data

In this section we describe CMD. We first present CAMELS, the simulation suite used to generate CMD. We then outline the different fields present in CMD. Next, we explain how the 2D maps and 3D grids were created. Finally, we discuss the labels, symmetries, and storage needs of CMD.

2.1. The CAMEL Simulations

CMD was generated from CAMELS data (Villaescusa-Navarro et al. 2021a). We now briefly describe those simulations here and refer the reader to Villaescusa-Navarro et al. (2021a) for further details.

CAMELS is a suite of 4233 numerical simulations: 2184 state-of-the-art (magneto)hydrodynamic simulations and 2049 gravity-only N-body simulations. All simulations follow the evolution of 256^3 dark matter particles plus 256^3 fluid elements (only in the case of hydrodynamic simulations) from $z = 127$ to the present time, $z = 0$, in a periodic volume of $(25\,h^{-1}\,\mathrm{Mpc})^3$. The initial conditions of all simulations were generated at redshift $z = 127$ using second-order Lagrangian perturbation theory (2LPT[29]). For simplicity, we assumed that the transfer functions of the dark matter and gas fluids of the (magneto) hydrodynamic simulations were the same and equal to the one of total matter. For each simulation, we saved snapshots at 34 different redshifts, from $z = 6$ to $z = 0$. To generate CMD, we used the snapshots at $z = [0, 0.5, 1, 1.5, 2]$.

While all simulations follow the evolution of dark matter particles, only the (magneto)hydrodynamic simulations solve the hydrodynamic equations and implement astrophysical effects such as star formation and feedback from stars and AGN. Each CAMEL simulation belongs to one of three suites: (1) IllustrisTNG (magnetohydrodynamic simulations), (2) SIMBA (hydrodynamic simulations), and (3) N-body (gravity-only simulations). CMD preserves this naming system. For instance, when addressing the IllustrisTNG neutral hydrogen maps, we refer to the 2D maps containing the neutral hydrogen

[25] See, e.g., https://github.com/georgestein/ml-in-cosmology for a comprehensive list of machine-learning applications in cosmology.

[26] http://yann.lecun.com/exdb/mnist/

[27] https://www.cs.toronto.edu/kriz/cifar.html

[28] https://image-net.org

[29] https://cosmo.nyu.edu/roman/2LPT/

field that were generated from the CAMELS IllustrisTNG simulation suite. We note that the IllustrisTNG and SIMBA simulations solve the hydrodynamic equations using two completely different methods and implement very different subgrid models.

All simulations share the values of the following cosmological parameters: $\Omega_b = 0.049$, $h = 0.6711$, $n_s = 0.9624$, $w = -1$, $\sum m_\nu = 0$, and $\Omega_K = 0$. The values of Ω_m and σ_8 are however varied across simulations.

The hydrodynamic simulations (IllustrisTNG and SIMBA) also vary four astrophysical parameters, coined $A_{\rm SN1}$, $A_{\rm SN2}$, $A_{\rm AGN1}$, and $A_{\rm AGN2}$ that control the efficiency of supernova and AGN feedback. It is important to emphasize that although the names of these parameters are the same in IllustrisTNG and SIMBA, their implementation and meaning are not identical. This should be taken into account while working with the feedback parameter labels (we provide more details on this in Section 2.5).

Each CAMEL simulation is thus characterized by the suite it belongs to (IllustrisTNG, SIMBA, or *N*-body) and either two or six numbers: two cosmological parameters (all simulations) and four astrophysical parameters (only IllustrisTNG and SIMBA). While the astrophysical parameters govern broadly similar quantities in IllustrisTNG and SIMBA, they are implemented differently, and hence their absolute values cannot be straightforwardly compared across models. Also, one cannot expect a one-to-one correspondence between the results of a simulation and the values of its parameters, as the simulations are also affected by cosmic variance: the finite volume that the simulations represent does not correspond to a representative sample of the whole universe. Thus, the link between quantities measured from a simulation and the values of its parameters can be considered as probabilistic.

The CAMELS simulations span a wide range in the values of the cosmological and astrophysical parameters. They were designed precisely in that way to avoid being affected by priors when training neural networks and also to allow overlap in parameter space when using different hydrodynamic simulations. On the one hand, we are mostly interested in performing inference on the value of the cosmological parameters; thus, many different models need to be simulated. On the other hand, we know that astrophysical effects can have an impact on cosmological observables, but we do not fully understand the physics of those effects, so the most conservative solution is to marginalize over them. For that reason, the values of the astrophysical parameters have to be varied over a wide range to perform a robust marginalization.

Each simulation suite contains four different sets:

1. LH (Latin-Hypercube) is a set of 1000 simulations for each of the IllustrisTNG and SIMBA suites, where the values of the cosmological and astrophysical parameters are sampled from a Latin-hypercube. Each of these simulations has a different initial random seed. The vast majority of 2D maps and 3D grids from CMD were generated from this simulation set. This is the main simulation set from which we build CMD.
2. 1P (one parameter at a time) is a set of 61 simulations with the same initial random seed but where only the value of one parameter is varied at a time. CMD contains data from this set.
3. CV (Cosmic Variance) is a set of 27 simulations with the same value of the cosmological and astrophysical parameters but different initial random seeds. A small fraction of 2D maps and 3D grids from CMD were generated from this set. The main purpose of the data from this set is to test the models trained on data from the LH set.
4. EX (Extreme) is a set of four simulations covering extreme models. CMD does not contain data from this set.

We now describe in more detail the particulars of each simulation type.

2.1.1. *IllustrisTNG*

The IllustrisTNG simulations have been run with the AREPO code[30] (Springel 2010) and employ the same subgrid physics as the original IllustrisTNG simulations (Weinberger et al. 2017; Pillepich et al. 2018b). In these simulations, $A_{\rm SN1}$ and $A_{\rm SN2}$ control two properties of galactic winds: the energy emitted per unit of star formation rate and the wind speed, respectively. $A_{\rm AGN1}$ and $A_{\rm AGN2}$ represent the energy released per unit of black hole accretion rate and the ejection speed (burstiness) for the kinetic mode of black hole feedback. We refer the reader to Villaescusa-Navarro et al. (2021a) for further details on these simulations and their astrophysical parameters.

2.1.2. *SIMBA*

The SIMBA simulations have been run with the GIZMO (Hopkins 2015) code[31] (Hopkins 2015) and employ the same subgrid physics as the original SIMBA simulation (Davé et al. 2019). The parameters $A_{\rm SN1}$ and $A_{\rm SN2}$ control the mass loading factor and speed of galactic winds relative to scalings derived from the FIRE simulations (Muratov et al. 2015; Anglés-Alcázar et al. 2017b). The parameter $A_{\rm AGN1}$ determines the momentum flux of kinetic outflows in quasar and jet-mode AGN feedback relative to the black hole accretion rate (Anglés-Alcázar et al. 2017a) while $A_{\rm AGN2}$ parameterizes the speed of the jet-mode black hole feedback. We refer the reader to Villaescusa-Navarro et al. (2021a) for further details on these simulations and their astrophysical parameters.

We note that although both the IllustrisTNG and SIMBA simulations aim at modeling the properties of cosmic gas, dark matter, and galaxies in a given cosmological model, the way they solve the hydrodynamic equations and implement their subgrid physics is substantially different. Thus, since neither simulation is a priori more accurate than another, this should also be seen as another factor to marginalize over when performing cosmological tasks such as parameter inference. For instance, for inference on the value of the cosmological parameters, it would be desirable that results do not depend on the simulation suite used for training a given model (see Section 3.1.4 for more details).

2.1.3. N-*body*

The gravity-only simulations have been run with the GADGET-III code (Springel 2005). They only follow the evolution of dark matter particles, which represent the cold dark matter plus baryon fluid, but as opposed to the above hydrodynamic simulations, they do not solve the hydrodynamic

[30] https://arepo-code.org/

[31] http://www.tapir.caltech.edu/~phopkins/Site/GIZMO.html

equations or model astrophysical effects such as supernova and AGN feedback. Thus, the data from these simulations is not affected by astrophysical effects, and therefore, CMD data created from these simulations can be seen as a pristine sample. These simulations only follow the evolution of the total matter field.

There is one *N*-body simulation for each hydrodynamical simulation in the IllustrisTNG and SIMBA suites. This is the reason why the number of 2D maps and 3D grids for the total matter field from these simulations is twice as large as those from the IllustrisTNG and SIMBA simulations. In other words, for each total matter map or grid, there is an *N*-body counterpart.

The *N*-body simulations use the same random amplitudes and phases for the initial perturbations as their hydrodynamical analogs, such that they follow "the same" universe, except for including only dark matter and not normal matter and correspondingly following only gravity and not other physics.

2.2. Fields

Before describing how we generate the CMD data from the CAMELS simulations, we first outline the different fields that comprise CMD.

The hydrodynamic simulations contain four different types of particles: (1) gas, (2) dark matter, (3) stars, and (4) black holes. The *N*-body simulations only have dark matter particles. All particle types have positions, velocities, and masses. The gas particles also contain a set of properties describing the physical state of the gas element they represent: pressure, temperature, metallicity, neutral hydrogen, electron number density, and magnetic fields (only in the case of IllustrisTNG simulations).

Each particle type represents a resolution element of its corresponding component, e.g., a gas particle represents a fluid element of cosmic gas. The actual 3D shape of these elements can be quite complicated[32] but we approximate them as uniform spheres for simplicity, characterized by their window function

$$W_{3\mathrm{D}}(\boldsymbol{x}) = \begin{cases} \dfrac{3}{4\pi R^3} & \text{if } |\boldsymbol{x}| \leqslant R \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

such that

$$\int_{\boldsymbol{x}} W_{3\mathrm{D}}(\boldsymbol{x})\, d^3\boldsymbol{x} = 1 \tag{2}$$

and where R is the radius of the sphere and the integral runs over all 3D space. In the case of gas and dark matter particles, we set this radius to the distance to the 32nd closest[33] gas and dark matter particles, respectively. For stars and black hole particles, we set $R = 0$, as the resolution of our 2D maps and 3D grids is too coarse to be able to resolve the internal structure of galaxies, i.e., the window function is just a Dirac delta.

We now describe in more detail each CMD field that represents the spatial distribution of a different property of gas, dark matter, and stars in a given universe. In the following equations for each field, the window function we refer to is the one in Equation (1) for 3D, and its projection

$$W_{2\mathrm{D}}(\boldsymbol{x}') = \int_z W_{3\mathrm{D}}(\boldsymbol{x})\, dz \tag{3}$$

when working in 2D (see Section 2.3 for further details). Table 1 summarizes the fields considered for each simulation suite.

1. *Gas density.* This field represents the spatial distribution of the density of cosmic gas. To construct 2D maps and 3D grids for this field, we need to read the positions and masses of all gas particles in a given simulation. For this field, the quantity stored in every pixel/voxel is the density of gas from all particles that contribute to that location, or more formally
$$\frac{1}{Q}\sum_i M_{\mathrm{g},i} \int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x} - \boldsymbol{r}_{\mathrm{g},i})\, d\boldsymbol{x}, \tag{4}$$
where $W_{\mathrm{g},i}$, $M_{\mathrm{g},i}$, and $\boldsymbol{r}_{\mathrm{g},i}$ are the window function, mass, and position of the gas particle i. The sum runs over all gas particles and the integral is over the area/volume of a given pixel/voxel. Q represents the area of a pixel in the case of 2D maps or the volume of a voxel for the 3D grids. This field is only present in the IllustrisTNG and SIMBA simulations.
2. *Gas velocity.* This field represents the spatial distribution of the modulus of the peculiar velocity vector of cosmic gas $v_\mathrm{g} = |\boldsymbol{v}_\mathrm{g}|$. To generate 2D maps and 3D grids for this field, we need to read the positions, masses, and velocities of all gas particles in a given simulation. The quantity stored in a pixel/voxel is the mass-weighted modulus of the gas velocity from all gas particles contributing to that location, or more formally
$$\frac{\sum_i M_{\mathrm{g},i} v_{\mathrm{g},i} \int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x} - \boldsymbol{r}_{\mathrm{g},i})\, d\boldsymbol{x}}{\sum_i M_{\mathrm{g},i} \int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x} - \boldsymbol{r}_{\mathrm{g},i})\, d\boldsymbol{x}}, \tag{5}$$
where $W_{\mathrm{g},i}$, $M_{\mathrm{g},i}$, $v_{\mathrm{g},i}$, and $\boldsymbol{r}_{\mathrm{g},i}$ are the window function, mass, velocity, and position of the gas particle i, respectively. The sum runs over all gas particles, and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG and SIMBA simulations.

   We note that the above quantity should not be seen as the gas bulk velocity, which should be computed as
$$\frac{\left|\sum_i M_{\mathrm{g},i} \boldsymbol{v}_{\mathrm{g},i} \int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x} - \boldsymbol{r}_{\mathrm{g},i})\, d\boldsymbol{x}\right|}{\sum_i M_{\mathrm{g},i} \int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x} - \boldsymbol{r}_{\mathrm{g},i})\, d\boldsymbol{x}}. \tag{6}$$
   Instead, in our definition, each gas particle makes a positive contribution to the quantity, and its magnitude will be larger for higher velocities, independently of their direction.
3. *Gas temperature.* This field represents the spatial distribution of the temperature of the cosmic gas. To generate 2D maps and 3D grids, we need to read the positions, masses, and temperatures of all gas particles in a given simulation. The gas temperature is computed as $T = (\gamma - 1)u/k_\mathrm{B}\mu$, where $\gamma = 5/3$, u is the internal energy, k_B is the Boltzmann constant, and μ is the mean

[32] For instance, in the case of the IllustrisTNG simulations, the gas particles are the mesh-generating points of a Voronoi tessellation of space.
[33] This number was chosen empirically as a compromise. Higher values will erase small-scale features and therefore produce a smoother field than the actual one. On the other hand, smaller values may give rise to sharp transitions and the appearance of noisy features.

Table 1
Number of 2D Maps and 3D Grids Available for the Different Fields and Simulations

			Simulation					
			IllustrisTNG		SIMBA		*N*-body	
Field	Prefix	Units	2D Maps	3D Grids	2D Maps	3D Grids	2D Maps	3D Grids
Gas density	Mgas	$(h^{-1}M_\odot)/(h^{-1}\text{ kpc})^A$	15,000	15,000	15,000	15,000	⋯	⋯
Gas velocity	Vgas	km s^{-1}	15,000	15,000	15,000	15,000	⋯	⋯
Gas temperature	T	K	15,000	15,000	15,000	15,000	⋯	⋯
Gas pressure	P	(km s^{-1})($M_\odot$ kpc^{-3})	15,000	15,000	15,000	15,000	⋯	⋯
Gas metallicity	Z	⋯	15,000	15,000	15,000	15,000	⋯	⋯
Neutral hydrogen density	H I	$h^{-1}M_\odot/(h^{-1}\text{ kpc})^A$	15,000	15,000	15,000	15,000	⋯	⋯
Electron number density	ne	$h^{-1}/(h^{-1}\text{ kpc})^A$	15,000	15,000	15,000	15,000	⋯	⋯
Magnetic fields	B	G	15,000	15,000	⋯	⋯	⋯	⋯
Magnesium over iron	MgFe	⋯	15,000	15,000	15,000	15,000	⋯	⋯
Dark matter density	Mcdm	$h^{-1}M_\odot/(h^{-1}\text{ kpc})^A$	15,000	15,000	15,000	15,000	⋯	⋯
Dark matter velocity	Vcdm	km s^{-1}	15,000	15,000	15,000	15,000	⋯	⋯
Stellar mass density	Mstar	$h^{-1}M_\odot/(h^{-1}\text{ kpc})^A$	15,000	15,000	15,000	15,000	⋯	⋯
Total matter density	Mtot	$h^{-1}M_\odot/(h^{-1}\text{ kpc})^A$	15,000	15,000	15,000	15,000	30,000*	30,000*
Total			195,000	195,000	180,000	180,000	30,000	30,000

Note. All 2D maps are at $z = 0$ and contain 256^2 pixels. The 3D grids contain 128^3, 256^3, or 512^3 voxels and are at redshifts $z = 0$, $z = 0.5$, $z = 1$, $z = 1.5$, or $z = 2$. We also quote the prefix used for the different fields together with the units used to store the information in the 2D maps and 3D grids. The exponent A that appears in the units of several densities has a value of 2 in the case of 2D maps and of 3 for 3D grids. The symbol * denotes that for each 2D map and 3D grid of the total matter field of the IllustrisTNG and SIMBA simulations, there is an *N*-body counterpart.

molecular weight. This expression is used for both gas and star-forming particles. The quantity stored in a pixel/voxel is the mass-weighted temperature of cosmic gas from all gas particles contributing to that location, or more formally

$$\frac{\sum_i M_{g,i} T_i \int_{\boldsymbol{x}} W_{g,i}(\boldsymbol{x} - \boldsymbol{r}_{g,i}) d\boldsymbol{x}}{\sum_i M_{g,i} \int_{\boldsymbol{x}} W_{g,i}(\boldsymbol{x} - \boldsymbol{r}_{g,i}) d\boldsymbol{x}} \quad (7)$$

where $W_{g,i}$, $M_{g,i}$, T_i, and $\boldsymbol{r}_{g,i}$ are the window function, mass, temperature, and position of the gas particle i, respectively. The sum runs over all gas particles and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG and SIMBA simulations.

4. *Gas pressure.* This field represents the spatial distribution of the pressure of the cosmic gas. To generate 2D maps and 3D grids, we need to read the positions, masses, and pressures of all gas particles in a given simulation. The gas pressure of each gas and star-forming particle is computed as $P = (\gamma - 1)u\rho$, where ρ is the gas density. The quantity stored in a pixel/voxel is the mass-weighted pressure of cosmic gas from all gas particles contributing to that location, or more formally

$$\frac{\sum_i M_{g,i} P_i \int_{\boldsymbol{x}} W_{g,i}(\boldsymbol{x} - \boldsymbol{r}_{g,i}) d\boldsymbol{x}}{\sum_i M_{g,i} \int_{\boldsymbol{x}} W_{g,i}(\boldsymbol{x} - \boldsymbol{r}_{g,i}) d\boldsymbol{x}} \quad (8)$$

where $W_{g,i}$, $M_{g,i}$, P_i, and $\boldsymbol{r}_{g,i}$ are the window function, mass, pressure, and position of the gas particle i. The sum runs over all gas particles, and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG and SIMBA simulations.

5. *Gas metallicity.* This field represents the spatial distribution of the metallicity of the cosmic gas. The metallicity is defined as the ratio between the mass in metals[34] and the total gas mass: $Z = M_{\rm metal}/M_g$. To generate 2D maps and 3D grids, we need to read the positions, masses, and metallicities of all gas particles in a given simulation. While IllustrisTNG and SIMBA follow the evolution of different metals, as described in Villaescusa-Navarro et al. (2021a), the gas metallicity is computed in the same manner in both simulations. We refer the reader to Nelson et al. (2019) and Davé et al. (2019) for further details on the metal enrichment method employed in these simulations. The quantity stored in a pixel/voxel is the mean metallicity of cosmic gas from all gas particles contributing to that location, or more formally

$$\frac{\sum_i M_{g,i} Z_i \int_{\boldsymbol{x}} W_{g,i}(\boldsymbol{x} - \boldsymbol{r}_{g,i}) d\boldsymbol{x}}{\sum_i M_{g,i} \int_{\boldsymbol{x}} W_{g,i}(\boldsymbol{x} - \boldsymbol{r}_{g,i}) d\boldsymbol{x}} \quad (9)$$

where $W_{g,i}$, $M_{g,i}$, Z_i, and $\boldsymbol{r}_{g,i}$ are the window function, mass, metallicity, and position of the gas particle i. The sum runs over all gas particles and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG and SIMBA simulations.

[34] In astronomy, metals are all elements heavier than hydrogen and helium.

6. *Neutral hydrogen density.* This field represents the spatial distribution of the density of neutral hydrogen. To generate 2D maps and 3D grids, we need to read the positions and neutral hydrogen masses of all gas particles in a given simulation. The neutral hydrogen mass of each gas particle is computed using the self-shielding fitting formula used in Rahmati et al. (2013), while we consider star-forming particles to be fully self-shielded against external radiation. We refer the reader to Villaescusa-Navarro et al. (2018). The quantity stored in a pixel/voxel is the total neutral hydrogen density from all gas particles contributing to that location, or more formally
$$\frac{1}{Q}\sum_i M_{\mathrm{H\,I},i}\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}, \tag{10}$$
where $W_{\mathrm{g},i}$, $M_{\mathrm{H\,I},i}$, and $\boldsymbol{r}_{\mathrm{g},i}$ are the window function, neutral hydrogen mass, and position of the gas particle i, respectively. The sum runs over all gas particles, and the integral is over the area/volume of the considered pixel/voxel. Q represents the area of a pixel in the case of 2D maps or the volume of a voxel for the 3D grids. This field is only present in the IllustrisTNG and SIMBA simulations.
7. *Electron number density.* This field represents the spatial distribution of the density of electrons. To generate 2D maps and 3D grids, we need to read the positions and electron abundances of all gas particles in a given simulation. The quantity stored in a pixel/voxel is the total electron number density from all gas particles contributing to that location, or more formally
$$\frac{1}{Q}\sum_i n_{e,i}\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}, \tag{11}$$
where $W_{\mathrm{g},i}$, $n_{e,i}$, and $\boldsymbol{r}_{\mathrm{g},i}$ are the window function, number of electrons, and position of the gas particle i, respectively. The sum runs over all gas particles and the integral is over the area/volume of the considered pixel/voxel. Q represents the area of a pixel in the case of 2D maps or the volume of a voxel for the 3D grids. This field is only present in the IllustrisTNG and SIMBA simulations.
8. *Magnetic fields.* This field represents the spatial distribution of the magnitude of the magnetic field vector of the cosmic gas: $B=|\boldsymbol{B}|$. To generate 2D maps and 3D grids, we need to read the positions, masses, and magnetic field modulus of all gas particles in a given simulation. The quantity stored in a pixel/voxel is the mass-weighted modulus of the magnetic field vector from all gas particles contributing to that location, or more formally
$$\frac{\sum_i M_{\mathrm{g},i}B_i\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}}{\sum_i M_{\mathrm{g},i}\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}} \tag{12}$$
where $W_{\mathrm{g},i}$, B_i, and $\boldsymbol{r}_{\mathrm{g},i}$ are the window function, magnetic field modulus, and position of the gas particle i. The sum runs over all gas particles, and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG simulations.
9. *Magnesium over iron ratio.* This field represents the spatial distribution of the ratio between the magnesium and iron masses of the cosmic gas. The mass of each gas particle represents the sum of the mass in hydrogen, helium, and metals of that particle. Among the metals, our simulations track the masses of the carbon, nitrogen, oxygen, magnesium, silicon, and iron elements. This field thus represents the ratio between the masses of those two elements that belong to each gas particle. To generate 2D maps and 3D grids, we need to read the positions, magnesium masses, and iron masses of all gas particles in a given simulation. The quantity stored in a pixel/voxel is the ratio between all magnesium and iron masses from all gas particles contributing to that location, or more formally
$$\frac{\sum_i M_{\mathrm{Mg},i}\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}}{\sum_i M_{\mathrm{Fe},i}\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}} \tag{13}$$
where $W_{\mathrm{g},i}$, $M_{\mathrm{Mg},i}$, $M_{\mathrm{Fe},i}$, and $\boldsymbol{r}_{\mathrm{g},i}$ are the window function, magnesium mass, iron mass, and position of the gas particle i, respectively. The sum runs over all gas particles, and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG and SIMBA simulations.
10. *Dark matter density.* This field represents the spatial distribution of the dark matter density. To generate 2D maps and 3D grids, we need to read the positions and masses of all dark matter particles in a given simulation. The quantity stored in a pixel/voxel is the dark matter density from all dark matter particles contributing to that location, or more formally
$$\frac{1}{Q}\sum_i M_{\mathrm{dm},i}\int_{\boldsymbol{x}} W_{\mathrm{dm},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{dm},i})d\boldsymbol{x}, \tag{14}$$
where $W_{\mathrm{dm},i}$, $M_{\mathrm{dm},i}$, and $\boldsymbol{r}_{\mathrm{dm},i}$ are the window function, mass, and position of the dark matter particle i, respectively. The sum runs over all dark matter particles, and the integral is over the area/volume of the considered pixel/voxel. Q represents the area of a pixel in the case of 2D maps or the volume of a voxel for the 3D grids. This field is only present in the IllustrisTNG and SIMBA simulations.[35]
11. *Dark matter velocity.* This field represents the spatial distribution of the modulus of the peculiar velocity vector of the dark matter $v_{\mathrm{dm}}=|\boldsymbol{v}_{\mathrm{dm}}|$. To generate 2D maps and 3D grids for this field, we need to read the positions, masses, and velocities of all dark matter particles in a given simulation. The quantity stored in a pixel/voxel is the mass-weighted modulus of the dark matter velocity from all dark matter particles contributing to that location, or more formally
$$\frac{\sum_i M_{\mathrm{dm},i}v_{\mathrm{dm},i}\int_{\boldsymbol{x}} W_{\mathrm{dm},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{dm},i})d\boldsymbol{x}}{\sum_i M_{\mathrm{dm},i}\int_{\boldsymbol{x}} W_{\mathrm{dm},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{dm},i})d\boldsymbol{x}}, \tag{15}$$
where $W_{\mathrm{dm},i}$, $M_{\mathrm{dm},i}$, $v_{\mathrm{dm},i}$, and $\boldsymbol{r}_{\mathrm{dm},i}$ are the window function, mass, velocity, and position of the dark matter particle i, respectively. The sum runs over all dark matter particles, and the integral is over the area/volume of the considered pixel/voxel. This field is only present in the IllustrisTNG and SIMBA simulations.

We note that the above quantity should not be seen

[35] Note, however, that for the N-body simulations, the definitions of the dark matter mass field and the total matter field (see below), which does exist for those simulations, are by construction identical.

as the dark matter bulk velocity, which should be computed as

$$\frac{\left|\sum_i M_{\mathrm{dm},i}\boldsymbol{v}_{\mathrm{dm},i}\int_{\boldsymbol{x}} W_{\mathrm{dm},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{dm},i})d\boldsymbol{x}\right|}{\sum_i M_{\mathrm{dm},i}\int_{\boldsymbol{x}} W_{\mathrm{dm},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{dm},i})d\boldsymbol{x}}. \tag{16}$$

Instead, in our definition, each gas particle has a positive contribution, and its magnitude will be larger for higher velocities, independently of their direction.

12. *Stellar mass density.* This field represents the spatial distribution of the stellar mass density. To generate 2D maps and 3D grids, we need to read the positions and masses of all stellar particles in a given simulation. The quantity stored in a pixel/voxel is the stellar mass density from all stellar particles contributing to that location, or more formally

$$\frac{1}{Q}\sum_i M_{*,i}\int_{\boldsymbol{x}} W_{*,i}(\boldsymbol{x}-\boldsymbol{r}_{*,i})d\boldsymbol{x}, \tag{17}$$

where $W_{*,i}$, $M_{*,i}$, and $\boldsymbol{r}_{*,i}$ are the window function, mass, and position of the stellar particle i, respectively. The sum runs over all stellar particles, and the integral is over the area/volume of the considered pixel/voxel. Q represents the area of a pixel in the case of 2D maps or the volume of a voxel for the 3D grids. This field is only present in the IllustrisTNG and SIMBA simulations.

13. *Total matter mass.* This field represents the spatial distribution of the total matter mass. Total matter is defined as the sum of gas, dark matter, stars, and black holes and thus represents the total amount of mass, baryonic as well as dark, in a given universe. To generate 2D maps and 3D grids, we need to read the positions and masses of all gas, dark matter, and stellar and black hole particles in a given simulation. The quantity stored in a pixel/voxel is the total matter density from all different particles contributing to that location, or more formally

$$\begin{aligned}\frac{1}{Q}\Bigg[&\sum_i M_{\mathrm{g},i}\int_{\boldsymbol{x}} W_{\mathrm{g},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{g},i})d\boldsymbol{x}\\&+\sum_i M_{\mathrm{dm},i}\int_{\boldsymbol{x}} W_{\mathrm{dm},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{dm},i})d\boldsymbol{x}\\&+\sum_i M_{*,i}\int_{\boldsymbol{x}} W_{*,i}(\boldsymbol{x}-\boldsymbol{r}_{*,i})d\boldsymbol{x}\\&+\sum_i M_{\mathrm{bh},i}\int_{\boldsymbol{x}} W_{\mathrm{bh},i}(\boldsymbol{x}-\boldsymbol{r}_{\mathrm{bh},i})d\boldsymbol{x}\Bigg],\end{aligned} \tag{18}$$

where $W_{\mathrm{g},i}$, $W_{\mathrm{dm},i}$, $W_{*,i}$, and $W_{\mathrm{bh},i}$ are the window function of i gas, dark matter, and stellar and black hole particles, respectively. $M_{\mathrm{g},i}$, $M_{\mathrm{dm},i}$, $M_{*,i}$, and $M_{\mathrm{bh},i}$ represent the mass of the i gas, dark matter, and stellar and black hole particles, correspondingly. $\boldsymbol{r}_{\mathrm{g},i}$, $\boldsymbol{r}_{\mathrm{dm},i}$, $\boldsymbol{r}_{*,i}$, and $\boldsymbol{r}_{\mathrm{bh},i}$ are the position of the i gas, dark matter, and stellar and black hole particles, respectively. The sums run over all particles of the different types, and the integral is over the area/volume of the considered pixel/voxel. Q represents the area of a pixel in the case of 2D maps or the volume of a voxel for the 3D grids.

We note that in the case of the N-body simulations, we only need to read the positions and masses of the dark matter particles and evaluate the second term of the above expression. This field is present in all three simulation types: IllustrisTNG, SIMBA, and N-body. We emphasize that the number of 2D maps and 3D grids from the N-body simulations is equal to the sum of those from the IllustrisTNG and SIMBA simulations, since for each map and grid from these simulations there is an N-body counterpart.

2.3. 2D Maps

We now describe the method we use to generate the 2D maps and their characteristics. In Table 1 we summarize the number and properties of the CMD 2D maps.

The 2D maps are generated as follows. First, we consider a given simulation and read the positions and properties of the considered field (see the above subsection for further details on each field). Next, we compute the radius of the considered particles as the distance to the 32nd closest gas and dark matter particle (in the case of gas and dark matter particles, respectively), or set it to zero for stellar and black hole particles. We then consider a slice of dimensions $25 \times 25 \times 5(h^{-1}\,\mathrm{Mpc})^3$ and select the particles that lie inside it. For each simulation, we take 15 slices: five in the XY plane, five in the XZ plane, and five in the YZ plane. We note that slices with the same projection direction do not overlap in space. We then project the window function of the considered particles into 2D:

$$W_{\mathrm{2D}}(r,\theta)=\int_z W_{\mathrm{3D}}(r,\theta,z)dz \tag{19}$$

where we have made use of cylindrical coordinates, and z represents the axes along which we project the slice. Note that by construction, the quantities will be preserved in the projection, i.e.,

$$\begin{aligned}\int_{\boldsymbol{x}'} W_{\mathrm{2D}}(r,\theta)d\boldsymbol{x}' &= \int_r\int_\theta W_{\mathrm{2D}}(r,\theta)drd\theta\\&=\int_r\int_\theta\int_z W_{\mathrm{3D}}(r,\theta,z)drd\theta dz=\int_{\boldsymbol{x}} W_{\mathrm{3D}}(r,\theta,z)d\boldsymbol{x}.\end{aligned} \tag{20}$$

Finally, we deposit the properties associated with these circles into a regular 2D grid with 256×256 pixels by performing the integrals of Section 2.2. We do this numerically, by sampling each circle with 1000 tracers that are distributed such that all of them cover the same area, and assign the contribution of each tracer to the pixel that contains its center. We note that in the case of particles that have a radius equal to 0 (i.e., stellar and black hole particles), we just assign the property of these particles to the pixel they belong to. Although this procedure is approximate, in the limit that the number of tracers tends to infinity, one recovers the correct exact result of depositing circles into a 2D regular grid. We have checked that with 1000 tracers our results are converged. We note that more accurate results can be obtained, if needed, from the 3D grids, whose constructions follow a very different procedure to the 2D maps (see Section 2.4 for more details). The reason why we describe the method used to generate the maps with this approximate procedure, instead of just saying that 2D slices can be taken from 3D cubes generated with a nonapproximated method, is because we want to allow any potential user to reproduce the results of Villaescusa-Navarro et al. (2021b, 2021c), whose networks were trained using these maps.

All 2D maps are created at $z = 0$ and contain 256×256 pixels over an area of 25×25 $(h^{-1}\,\mathrm{Mpc})^2$. For each field, we generate 15,000 maps: 15 maps per simulation for 1000 simulations. Each 2D map is described by either two or six numbers: two cosmological parameters (all maps) and four astrophysical parameters (only the maps generated from the IllustrisTNG and SIMBA simulations).

The reason why CMD only contains 2D maps at $z = 0$ is because it is possible to generate maps from the 3D grids (see Section 2.4), and also to allow users to fully reproduce the results obtained for the parameter inference task described below. For instance, one can take a slice of a 3D grid and project it into a 2D plane. From the 3D grids, we can generate 2D maps with different widths, at different resolutions, and different redshifts. Furthermore, in some cases, one may want to use 2D maps that partially overlap in the projected direction. We thus recommend readers use the 3D grids to generate 2D maps that fulfill their needs.

We note that different fields from the same simulation may exhibit a very tight spatial correlation on large scales. This can be seen in Figure 1, where we show an example of CMD maps of the IllustrisTNG simulations for all fields. In the online documentation we show examples of how to read and manipulate the files containing the 2D maps. We also describe there how to generate 2D maps from the 3D grids.

2.4. 3D Grids

We now describe the method we employ to generate the 3D grids and present their features. Table 1 shows the characteristics of the CMD 3D grids.

The 3D grids are constructed as follows. First, we read the positions and properties of the considered particles (see Section 2.2 for details). Next, we compute the radius of the considered particles as the distance to the 32nd closest gas and dark matter particle (in the case of gas and dark matter particles, respectively), or set it to zero for stellar and black hole particles. Next, we deposit the properties sampled by the spheres into a regular 3D grid with 128^3, 256^3, or 512^3 voxels using VOXELIZE,[36] a code that calculates the integrals of Section 2.2 in a precise way, making use of the OVERLAP code (Strobl et al. 2016).[37] We emphasize that the window function of the particles is given by Equation (1) and is not the same for different particles. This means that the window function cannot be trivially removed from power spectra in order to enable direct comparison to theory predictions. However, for machine-learning applications, this is not an issue, and we believe that the choice of spherical kernels with physically motivated radii leads to smoother fields, which may benefit training of convolutional neural networks (CNNs).

The 3D grids are constructed at redshifts 0, 0.5, 1, 1.5, and 2. For each simulation, field, and redshift, CMD provides three grids with 128^3, 256^3, and 512^3 voxels. We note that these three grids represent exactly the same field at different resolutions. All grids cover a co-moving periodic volume of $(25\,h^{-1}\,\mathrm{Mpc})^3$. Different fields from the same simulation may exhibit a very tight spatial correlation on large scales, as in the case of the 2D maps.

[36] https://github.com/leanderthiele/voxelize

[37] https://github.com/severinstrobl/overlap

In the online documentation, we provide examples illustrating how to read the files containing the 3D fields and how to manipulate the data.

2.5. Labels

The 2D maps and 3D grids from the IllustrisTNG and SIMBA simulations are characterized by six numbers: two cosmological parameters (Ω_m and σ_8), and four astrophysical parameters (A_SN1, A_SN2, A_AGN1, and A_AGN2). In the case of maps and grids from the N-body simulations, the labels are only Ω_m and σ_8. By construction, each parameter can vary within a wide range; many values are so extreme that the corresponding universes simulated in CAMELS are far from reality. This is however not a problem, as we want our results to be unaffected by our priors (Villaescusa-Navarro et al. 2020). In Table 2 we show the range of variation of each parameter together with its distribution.

While the values of the cosmological parameters represent the same property in all simulations, 2D maps, and 3D grids, the same is not true for the astrophysical parameters, whose definition and implementation is different in the IllustrisTNG and SIMBA simulations. Thus, if a neural network is trained to infer the value of Ω_m and σ_8 from 2D maps of IllustrisTNG simulations, the same network can be tested on 2D maps from the SIMBA and N-body simulations to see if it is able to recover the values of those parameters. On the other hand, a network trained on, e.g., SIMBA 3D grids to infer the values of the astrophysical parameters, is not expected to work when tested in 3D grids from the IllustrisTNG simulations.

For cosmological applications, in general, the astrophysical parameters and their effects should be taken as nuisance parameters to be marginalized over.

2.6. Symmetries

It is important to know the symmetries of the CMD data to exploit them, or to enforce them, when using machine-learning models or other methods.

The simulations are equivariant under rotations around integer multiples of $\pi/2$ (equivariance for other rotation angles holds approximately), parity, and translations. In order to implement the latter, periodic boundary conditions apply (not for cropped regions, of course). This information can be useful in some cases. For instance, instead of padding convolutional layers with zeros, one can use periodic padding for the same task, which may improve the performance of the model.

2.7. Data Volume

We now briefly describe the disk space needed to store the different CMD elements. The property stored in every pixel of a 2D map and in every voxel of a 3D grid is a float that takes 4 bytes. Thus, a 2D map occupies $256 \times 256 \times 4 = 0.25$ MB. In CMD, every field map contains 15,000 maps, so these files will require 3.7 GB per field. Since CMD has 27 different files for the 2D maps (13 fields for IllustrisTNG, 12 fields for SIMBA, and 1+1 field for N-body[38]), storing all CMD 2D maps will require $\sim$100 GB.

[38] We do not store the 30,000 maps in a single file, but in two, in order to facilitate the correspondence with the maps from the IllustrisTNG and SIMBA maps.

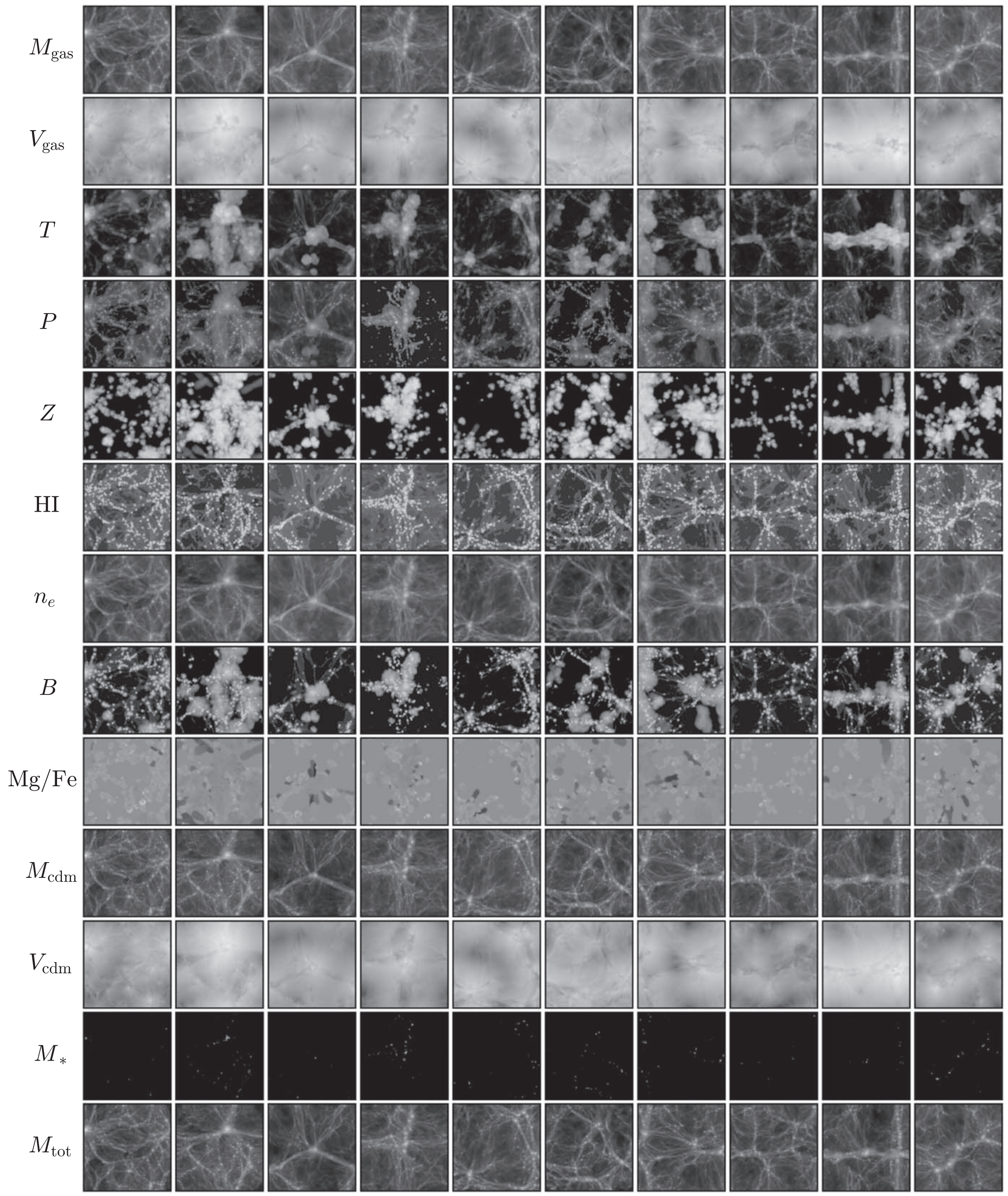

Figure 1. We show 10 examples of 2D maps for each of the 13 different fields present in the IllustrisTNG simulations. Each image has a different value of the cosmological parameters, $\Omega_{\rm m}$ and σ_8, and astrophysical parameters $A_{\rm SN1}$, $A_{\rm SN2}$, $A_{\rm AGN1}$, and $A_{\rm AGN2}$. The different columns represent the same region for the different fields. Each image has a physical size of $25 \times 25 \ (h^{-1}\,{\rm Mpc})^2$.

Table 2
Each 2D Map and 3D Grid of CMD Is Characterized by Six Numbers (Two in the Case of Data from the *N*-body Simulations)

Parameter	Range of Variation	Distribution
Ω_m	0.1–0.5	uniform
σ_8	0.6–1.0	uniform
A_{SN1}	0.25–4.0	log uniform
A_{SN2}	0.5–2.0	log uniform
A_{AGN1}	0.25–4.0	log uniform
A_{AGN2}	0.5–2.0	log uniform

Note. This table shows the range of variation and the distribution of each parameter. "log uniform" means that the distribution of the parameter is uniform when the logarithm of the value is taken.

The files hosting the 3D grids instead contain 1000 grids per field, so each of those field files will occupy $N^3 \times 4 \times 1000$ bytes, i.e., 7.8 GB ($N = 128$), 62.5 GB ($N = 256$), and 500 GB ($N = 512$). Hence, the 27 field files and three different resolutions combined will require 15 TB for all 3D grids at a single redshift. All CMD grids, at the five different redshifts require 75 TB.

We note that in the future, we will generate more 2D maps and 3D grids at additional redshifts. The online documentation will always be updated to reflect any new data that is not described in this paper.

3. Applications

In this section we outline a few possible tasks that can be carried out with CMD. The list of applications discussed below is far from comprehensive, but is rather just a subset of all possible applications that can be carried out with such a rich and complex data set.

3.1. Parameter Inference

One of the main applications of CMD is parameter inference: given a 2D map or a 3D grid, $\boldsymbol{X}$, develop a method that predicts the posterior of the parameters

$$p(\boldsymbol{\theta}|\boldsymbol{X}) \tag{21}$$

where $\boldsymbol{\theta}$ can be a single parameter, e.g., Ω_m, or several or them such as $\boldsymbol{\theta} = \{\Omega_m, \sigma_8\, A_{SN1}, A_{SN2}, A_{AGN1}, A_{AGN2}\}$. We emphasize that due to cosmic variance, i.e., due to the finite volume covered in the simulations, no one-to-one correspondence between the 2D maps or 3D grids and the values of the parameters exists. The inference can be carried out from a single field, or several fields can be used together in what is called a *multifield*.

Being able to extract cosmological information, at the field level, from 2D maps and 3D grids while marginalizing over astrophysical effects is one of the main goals of modern cosmology. CMD represents a state-of-the-art data set that is optimized for this task. In our companion papers (Villaescusa-Navarro et al. 2021b, 2021c), we have made use of CMD to perform this task for the first time. Here, we now describe in detail the architecture and training procedure used in those works, and describe the problems we encountered as a challenge to the community. We note that in those works, we only used the 2D maps, so all of the details below apply to this data and not to the 3D grids. We also make publicly available all codes and network weights for this task at https://camels-multifield-dataset.readthedocs.io.

The way we carried out the inference in our companion papers (Villaescusa-Navarro et al. 2021b, 2021c) was to predict the mean and standard deviation of the marginal posterior for each parameter. Thus, given a 2D map, the network will output two numbers for each parameter.

3.1.1. Architecture

The architecture of our model consists of a set of convolutional layers (CNNs) followed by a fully connected layer. In detail:

1. **Input:** $C \times 256 \times 256 \rightarrow$

2. CNN (3, 1, 1) $\rightarrow 2H \times 256 \times 256 \rightarrow$
3. LeakyReLU $\rightarrow$
4. CNN(3,1,1) $\rightarrow 2H \times 256 \times 256 \rightarrow$
5. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
6. CNN(2,2,0) $\rightarrow 2H \times 128 \times 128 \rightarrow$
7. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

8. CNN(3,1,1) $\rightarrow 4H \times 128 \times 128 \rightarrow$
9. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
10. CNN(3,1,1) $\rightarrow 4H \times 128 \times 128 \rightarrow$
11. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
12. CNN(2,2,0) $\rightarrow 4H \times 64 \times 64 \rightarrow$
13. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

14. CNN(3,1,1) $\rightarrow 8H \times 64 \times 64 \rightarrow$
15. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
16. CNN(3,1,1) $\rightarrow 8H \times 64 \times 64 \rightarrow$
17. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
18. CNN(2,2,0) $\rightarrow 8H \times 32 \times 32 \rightarrow$
19. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

20. CNN(3,1,1) $\rightarrow 16H \times 32 \times 32 \rightarrow$
21. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
22. CNN(3,1,1) $\rightarrow 16H \times 32 \times 32 \rightarrow$
23. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
24. CNN(2,2,0) $\rightarrow 16H \times 16 \times 16 \rightarrow$
25. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

26. CNN(3,1,1) $\rightarrow 32H \times 16 \times 16 \rightarrow$
27. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
28. CNN(3,1,1) $\rightarrow 32H \times 16 \times 16 \rightarrow$
29. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
30. CNN(2,2,0) $\rightarrow 32H \times 8 \times 8 \rightarrow$
31. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

32. CNN(3,1,1) $\rightarrow 64H \times 8 \times 8 \rightarrow$
33. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
34. CNN(3,1,1) $\rightarrow 64H \times 8 \times 8 \rightarrow$
35. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$
36. CNN(2,2,0) $\rightarrow 64H \times 4 \times 4 \rightarrow$
37. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

38. CNN(4,1,0) $\rightarrow 128H \times 1 \times 1 \rightarrow$
39. BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$

40. Flatten tensor $\rightarrow 128H \rightarrow$
41. Dropout (DR) $\rightarrow$
42. FC(128H, 64H) $\rightarrow 64H$
43. LeakyReLU $\rightarrow$ Dropout (DR) $\rightarrow$
44. FC(64H, 12) $\rightarrow$ 12

45. **Output:** mean + std posterior (12 numbers)

where C is the number of channels of the input map; for single fields, $C = 1$, while this number is larger than one in the case of a multifield. H is a hyperparameter that controls the number of channels in the different CNNs; larger values will increase the number of channels and therefore make the network more complex. "DR" is the dropout rate, which is another hyperparameter of the network. The notation CNN(K,S,P) indicates a CNN layer with kernel size K, strides S, and padding P. The input and output number of channels can be inferred from the scheme. For example, the fourth CNN (step 8 above) takes as input $2H$ channels of images having 128×128 pixels and returns $4H$ channels of images that have 128×128 pixels. All CNN layers use periodic padding.

FC(A,B) indicates a fully connected layer with A and B input and output values, respectively. The model has 12 outputs, corresponding to the mean and standard deviation of the marginal posterior for each of the six cosmological and astrophysical parameters. In the case of N-body maps, we set the number of output features to 4, since these maps are fully characterized by $\Omega_{\rm m}$ and σ_8.

3.1.2. Loss Function

The above model outputs two numbers per parameter, the mean (μ_i) and standard deviation (σ_i) of the marginal posterior:

$$\mu_i(\boldsymbol{X}) = \int_{\theta_i} p(\theta_i|\boldsymbol{X})\theta_i d\theta_i, \tag{22}$$

$$\sigma_i(\boldsymbol{X}) = \int_{\theta_i} p(\theta_i|\boldsymbol{X})(\theta_i - \mu_i)^2 d\theta_i, \tag{23}$$

where $p(\theta_i|\boldsymbol{X})$ is the marginal posterior over the parameter i

$$p(\theta_i|\boldsymbol{X}) = \int_{\theta} p(\theta_1, \theta_2, \ldots \theta_n|\boldsymbol{X}) d\theta_1 \ldots d\theta_{i-1} d\theta_{i+1} \ldots d\theta_n. \tag{24}$$

In this notation, $\boldsymbol{X}$ represents a 2D map. Following the moments network work presented in Jeffrey et al. (2021), we define the loss function such that the output of the network converges to the above quantities:

$$\mathcal{L} = \sum_{i=1}^{6} \log\left(\sum_{j\in \rm batch} (\theta_{i,j} - \mu_{i,j})^2\right) + \sum_{i=1}^{6} \log\left(\sum_{j\in \rm batch} ((\theta_{i,j} - \mu_{i,j})^2 - \sigma_{i,j}^2)^2\right). \tag{25}$$

We note that this loss function differs from the original one presented in Jeffrey et al. (2021),

$$\mathcal{L} = \sum_{i=1}^{6} \sum_{j\in \rm batch} ((\theta_{i,j} - \mu_{i,j})^2 + ((\theta_{i,j} - \mu_{i,j})^2 - \sigma_{i,j}^2)^2). \tag{26}$$

We replace the arithmetic sum by the sum of the logarithm of each term (either posterior mean or posterior standard deviation). Empirically, we have found that our modified loss function provides much tighter and reliable values for μ_i and σ_i than its original version.

The reason for this is that different parameters can be constrained more accurately than others by the network. For instance, there are fields where the impact of astrophysical processes is very mild, while they are very sensitive to cosmology (e.g., the dark matter field). In those cases, the loss function in Equation (26) will be eventually dominated by the contribution from the astrophysical parameters, preventing the network from further improving constraints on the cosmological parameters.

By taking the logarithm of each term, before the sum, as in Equation (25), we are effectively multiplying the losses of all terms instead of summing them, which prevents the problem mentioned above from occurring. A different way to see this is that when taking the logarithm of the terms, we are rescaling all terms to the same order of magnitude, and therefore the loss function will give a similar weight to all terms.

3.1.3. Training Procedure

Each field contains 15,000 2D maps that we split into training (13,500 maps), validation (750 maps), and testing (750 maps) sets. This split is not done randomly from the maps themselves, but rather based on the simulations they were generated from, such that maps that have the same value of the cosmological and astrophysical parameters all belong to only one of these three sets. Thus, we take maps from 900, 50, and 50 simulations for training, validation, and testing, respectively, which amounts to the numbers of maps above given that there exist 15 maps for each simulation. We do this in this way to avoid hidden correlations between maps from the same simulation that we do not want the network to learn. The maps are then normalized as follows. First, we take the logarithm of the pixel values of all maps (we do $\log(1 + \rho_*)$ in the case of the stellar mass density maps). Next, we compute the mean and standard deviation from all pixels in all maps. Next, we normalize each single map by subtracting the mean and dividing by the standard deviation calculated before. We note that this procedure is also carried out when working with multifields. In other words, in the case of multifields, each channel/field can have a very different range (can be orders of magnitude) when compared to other channel/field. Thus, we normalize each channel individually.

Our model has four hyperparameters: (1) the learning rate (lr), (2) the weight decay (wd), (3) the number of channels in the CNNs (H), and (4) the dropout rate of the fully connected layers (DR). For a given value of the hyperparameters, we train the above network by minimizing the loss function of Equation (25) using gradient descent. We employ the AdamW optimizer (Loshchilov & Hutter 2017) with betas equal to 0.5 and 0.999.[39]

We train using a cyclicLR scheduler (Smith 2015) with a minimum learning rate of 10^{-9} and a maximum equal to lr. We take 500 steps up and another 500 steps down to define the period of the scheduler. We use a batch size of 128 and train the model for 200 epochs. We save the weights of the model with the lowest validation loss.

The value of the hyperparameters is optimized using the OPTUNA package (Akiba et al. 2019). For each field, we consider at least 50 trials, where a trial represents the result of training the network with a given value of the hyperparameters. Thus, for each field, we save the weights of at least 50 different models.

OPTUNA produces a database with the information of every trial, such as trial number, value of the hyperparameters, validation loss, etc. We further explain how to read these databases together with the files containing the weights of the

[39] See, e.g., Goodfellow et al. (2016), for more information on these and other common deep-learning concepts.

networks. With those files in hand, the model can be tested on either the test set or a new data set.

3.1.4. Challenges

In our companion papers (Villaescusa-Navarro et al. 2021b, 2021c), we show that the above architecture allows us to place a few percent constraints on the value of the cosmological parameters from almost all of the 13 different fields. However, we also encountered a series of obstacles and leave some work for future exploration. Here we outline what we believe are the most important challenges when doing parameter inference from CMD.

First, while our model is able to infer the value of the cosmological and astrophysical parameters with high accuracy for all of the fields, the question remains of whether ours is the best model that can be constructed. It would be interesting to find models that perform better, i.e., that can constrain the values of the parameters with higher accuracy. Knowing these constraints will allow us to better understand a deep and crucial question in cosmology: How much information do nonlinear Gaussian fields contain? The codes and weights released in this work can be taken as a benchmark to improve upon.

Second, we found that for some fields, e.g., gas temperature, if we train the model using IllustrisTNG maps and test it on SIMBA maps (and vice-versa), it is not able to recover the true value of the cosmological parameters. It is crucial for the model to be robust to the training data since models trained on simulated data may always face this problem, as simulations may never be perfect representations of reality. Thus, the second challenge is to find *robust* models that can be trained on a given simulation data set and will be able to perform inference when testing on a different data set.

Third, in our companion paper (Villaescusa-Navarro et al. 2021b), we found that when doing parameter inference over 2D maps with different fields as different channels (*multifield*), the constraints on the parameters improve with respect to a single field. However, the improvement is relatively modest, i.e., we did not observe a major improvement. Thus, it will be interesting to find the minimum set of fields of a multifield that contain, e.g., 90% and 95% of the cosmological and possibly astrophysical information. In other words, if using the 13 different fields allows us to place a given constraint on the value of the cosmological and astrophysical parameters, what minimum subset of these fields enables us to get a fraction of those constraints? The idea behind this exercise is that those subsets of fields can be used to infer the value of the cosmological and astrophysical parameters, and the rest of the fields can be used to perform internal cross-validations. For instance, if the gas temperature field is not used to infer the parameter values, it still can be used when running simulations with the most likely parameter values to compare directly against observations, providing an additional internal check to verify the robustness of the model. In other words, the temperature field from simulations that are run with the most probable value of the cosmological and astrophysical parameters can be used as a direct theoretical prediction to compare with actual observations.

Fourth, it would be very important to understand where the information from the different fields is coming from. In other words, what summary statistic, if any, are the CNNs using in order to constrain the value of the cosmological parameters? Are they focusing their attention on regions with large values of the considered field? Shedding light onto this question will help to develop analytic methods to more robustly extract that information but also help us in understanding the process of nonlinear gravitational evolution.

Of course, the above challenges apply to both 2D maps and 3D grids.

3.2. Generative Models

While CMD data spans a wide range of the values of cosmological and astrophysical parameters, there may be some applications where more data is needed at points in parameter space not covered by CMD data. In this case, one can use techniques such as conditional generative adversarial networks or conditional normalizing flows to generate new 2D maps or 3D grids conditioned on the values of the parameters (Tamosiunas et al. 2021). These emulators at the field level can be used in place of the more expensive simulations to carry out different tasks.

3.3. N*-body to Hydrodynamic*

(Magneto)hydrodynamic simulations are much more computationally expensive than gravity-only N-body simulations. At the time of writing this paper, running full hydrodynamic simulations over gigaparsec volumes with a reasonable resolution is computationally unfeasible. On the other hand, the N-body counterparts of these simulations are presently beginning to be feasible with advanced supercomputers. Thus, it may be desirable to *paint* gas and stellar properties onto the dark matter field from N-body simulations (i.e., to transform dark matter into gas and stellar properties), as in Wadekar et al. (2020, 2021), Thiele et al. (2020), Zhang et al. (2019), Yip et al. (2019), Kasmanoff et al. (2020), Harrington et al. (2021), and Jo & Kim (2019). Alternatively, the total matter field from the N-body simulations can be mapped to the total matter of the full hydrodynamic simulation. This will be necessary for creating weak lensing maps that incorporate astrophysics effects at the field level.

Since the CMD spans a large volume in parameter space, the above mapping(s) can be done conditionally on the values of the cosmological and astrophysical parameters. Furthermore, the mapping can be done for several fields at the same time in order to take into account all cross-correlations among the fields.

3.4. Superresolution

CMD provides 3D grids at three different resolutions for 13 different fields. It would be very interesting to train models that can take as input the low-resolution map or grid from a given field and output a higher-resolution version of it. We emphasize that the three different grids for each redshift and field provided by CMD arise from the same simulation, i.e., the mass and spatial resolution of the underlying simulation is the same.

Ideally, one would like to run a low-resolution hydrodynamic simulation and use a model that can produce a higher-resolution version of it. This will be extremely valuable for the astrophysics community as the computational cost quickly increases with mass and spatial resolution. While this task has been carried out for gravity-only simulations (Kodi Ramanah et al. 2020; Li et al. 2021; Ni et al. 2021), it still remains to be performed for cosmological hydrodynamic simulations. While the only difference between the three different CMD 3D grids is in the resolution of the mesh size rather than the resolution of the underlying simulation, developing superresolution methods for fields from cosmological hydrodynamical simulations using CMD has the potential to contribute toward the development of such methods for simulations of different resolutions.

3.5. Time Evolution

With enough disk space, it is possible to save as many snapshots of a simulation as desired. In practice, the number of snapshots generated is limited due to disk space constraints. It would be very valuable to have a model that, given a set of snapshots at some redshifts, could output snapshots at other redshifts. This will be valuable for understanding the time evolution of some phenomena and for constructing merger trees and lightcones from the simulations. Chen et al. (2020) showed that this is possible for gravity-only simulations. CMD provides a rich data set to train models to carry out this task for many different fields of hydrodynamic simulations.

4. Summary

In this paper we have introduced the CAMELS CMD, a large cosmological and astrophysical data set that contains hundreds of thousands of 2D maps and 3D grids of 13 different fields: (1) gas mass, (2) gas velocity, (3) gas temperature, (4) gas pressure, (5) gas metallicity, (6) neutral hydrogen mass, (7) electron density, (8) magnetic fields, (9) magnesium over iron ratio, (10) dark matter mass, (11) dark matter velocity, (12) stellar mass, and (13) total matter mass. CMD has been created from simulations of the CAMELS project (Villaescusa-Navarro et al. 2021a), a collection of more than 4000 gravity-only *N*-body simulations and state-of-the-art hydrodynamic simulations from thousands of different universes. Each 2D map and 3D grid is described by two cosmological and four astrophysical parameters (only in the case of the hydrodynamic simulations).

We have described a few applications of CMD: (1) parameter inference, (2) generative models, (3) mapping *N*-body to hydrodynamic, (4) superresolution, and (5) time evolution. In our companion papers (Villaescusa-Navarro et al. 2021b, 2021c), we used CMD to show that neural networks can extract information from the field while marginalizing over astrophysical effects for all CMD fields. In this paper we have described in detail the architecture of our model together with the training procedure.

We release all CMD data (over 70 TB), together with the codes and network weights for the parameter inference task carried out in our companion papers (Villaescusa-Navarro et al. 2021b, 2021c). We provide further technical details on how to download, read, and manipulate the data in https://camels-multifield-dataset.readthedocs.io. We hope that CMD can become a standard data set for machine-learning applications in cosmology and astrophysics.

In the future, we will create maps for directly observable quantities from weak-lensing, thermal and kinetic Sunyaev–Zeldovich effects, and X-ray and 21 cm emission. Future public data releases from the CAMELS simulation suite will also include a database specifically targeted at multiwavelength probes of circumgalactic medium profiles.

We thank the referee for his/her constructive report. F.V.N. acknowledges funding from the WFIRST program through NNG26PJ30C and NNN12AA01C. D.A.A. was supported in part by NSF grants AST-2009687 and AST-2108944. The work of D.N.S., S.G., D.A.A., L.T., Y.L., A.N., S.H., and B.D. W. has been supported by the Simons Foundation. The work of P.V.D. is supported by CIDEGENT/2018/019, CPI-21-108. The 2D maps and 3D grids have been created using the Pylians3 libraries and voxelize. CMD has been created using the Rusty cluster of the Flatiron Institute. Details on the CAMELS simulations can be found in https://www.camel-simulations.org.

Software: AREPO (Springel 2010), Gadget-III (Springel 2005), GIZMO (Hopkins 2015), Optuna (Akiba et al. 2019), Overlap (Strobl et al. 2016), Pylians (https://github.com/franciscovillaescusa/Pylians3), Voxelize (https://github.com/leanderthiele/voxelize).

ORCID iDs

Francisco Villaescusa-Navarro https://orcid.org/0000-0002-4816-0455
Shy Genel https://orcid.org/0000-0002-3185-1540
Daniel Anglés-Alcázar https://orcid.org/0000-0001-5769-4945
Leander Thiele https://orcid.org/0000-0003-2911-9163
Romeel Dave https://orcid.org/0000-0003-2842-9434
Desika Narayanan https://orcid.org/0000-0002-7064-4309
Yin Li https://orcid.org/0000-0002-0701-1410
Pablo Villanueva-Domingo https://orcid.org/0000-0002-0936-4279
Benjamin Wandelt https://orcid.org/0000-0002-5854-8269
David N. Spergel https://orcid.org/0000-0002-5151-0006
Faizan G. Mohammad https://orcid.org/0000-0001-9243-7434
Sultan Hassan https://orcid.org/0000-0002-1050-7572
Helen Shao https://orcid.org/0000-0002-0152-6747
Digvijay Wadekar https://orcid.org/0000-0002-2544-7533
Emily Moser https://orcid.org/0000-0003-1593-1505
Erwin T. Lau https://orcid.org/0000-0001-8914-8885
Lucia A. Perez https://orcid.org/0000-0002-8449-1956
Daisuke Nagai https://orcid.org/0000-0002-6766-5942
Nicholas Battaglia https://orcid.org/0000-0001-5846-0411
Mark Vogelsberger https://orcid.org/0000-0001-8593-7692

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. 2019, arXiv:1907.10902
Alves de Oliveira, R., Li, Y., Villaescusa-Navarro, F., Ho, S., & Spergel, D. N. 2020, arXiv:2012.00240
Anglés-Alcázar, D., Davé, R., Faucher-Giguère, C.-A., Özel, F., & Hopkins, P. F. 2017a, MNRAS, 464, 2840
Anglés-Alcázar, D., Faucher-Giguère, C.-A., Kereš, D., et al. 2017b, MNRAS, 470, 4698
Böhm, V., & Seljak, U. 2020, arXiv:2006.05479
Chen, C., Li, Y., Villaescusa-Navarro, F., Ho, S., & Pullen, A. 2020, arXiv:2012.05472
Cybenko, G. 1989, Math. Control Signals Systems, 2, 303
Dai, B., & Seljak, U. 2021, PNAS, 118, 2020324118
Davé, R., Anglés-Alcázar, D., Narayanan, D., et al. 2019, MNRAS, 486, 2827
Genel, S., Vogelsberger, M., Springel, V., et al. 2014, MNRAS, 445, 175
Giusarma, E., Reyes Hurtado, M., Villaescusa-Navarro, F., et al. 2019, arXiv:1910.04255
Goodfellow, I., Bengio, Y., & Courville, A. 2016, Deep Learning (Cambridge, MA: MIT Press)
Harrington, P., Mustafa, M., Dornfest, M., Horowitz, B., & Lukić, Z. 2021, arXiv:2106.12662

He, S., Li, Y., Feng, Y., et al. 2019, PNAS, 116, 13825
Hopkins, P. F. 2015, MNRAS, 450, 53
Hornik, K. 1991, NN, 4, 251
Hornik, K., Stinchcombe, M., & White, H. 1990, NN, 3, 551
Jeffrey, N., Alsing, J., & Lanusse, F. 2021, MNRAS, 501, 954
Jo, Y., & Kim, J.-H 2019, MNRAS, 489, 3565
Kasmanoff, N., Villaescusa-Navarro, F., Tinker, J., & Ho, S. 2020, arXiv:2012.00186
Kodi Ramanah, D., Charnock, T., Villaescusa-Navarro, F., & Wandelt, B. D. 2020, MNRAS, 495, 4227
Li, Y., Ni, Y., Croft, R. A. C., et al. 2021, PNAS, 118, 2022038118
Loshchilov, I., & Hutter, F. 2017, arXiv:1711.05101
Makinen, T. L., Lancaster, L., Villaescusa-Navarro, F., et al. 2021, JCAP, 2021, 081
Modi, C., Feng, Y., & Seljak, U. 2018, JCAP, 2018, 028
Moews, B., Davé, R., Mitra, S., Hassan, S., & Cui, W. 2021, MNRAS, 504, 4024
Muratov, A. L., Kereš, D., Faucher-Giguère, C.-A., et al. 2015, MNRAS, 454, 2691
Nelson, D., Springel, V., Pillepich, A., et al. 2019, ComAC, 6, 2
Ni, Y., Li, Y., Lachance, P., et al. 2021, MNRAS, 507, 1021
Pillepich, A., Nelson, D., Hernquist, L., et al. 2018a, MNRAS, 475, 648
Pillepich, A., Springel, V., Nelson, D., et al. 2018b, MNRAS, 473, 4077
Rahmati, A., Pawlik, A. H., Raicevic, M., & Schaye, J. 2013, MNRAS, 430, 2427
Smith, L. N. 2015, arXiv:1506.01186
Somerville, R. S., & Davé, R. 2015, ARA&A, 53, 51
Springel, V. 2005, MNRAS, 364, 1105
Springel, V. 2010, MNRAS, 401, 791
Storey-Fisher, K., Huertas-Company, M., Ramachandra, N., et al. 2021, MNRAS, 508, 2946
Strobl, S., Formella, A., & Pöschel, T. 2016, JCoPh, 311, 158
Tamosiunas, A., Winther, H. A., Koyama, K., et al. 2021, MNRAS, 506, 3049
Thiele, L., Villaescusa-Navarro, F., Spergel, D. N., Nelson, D., & Pillepich, A. 2020, ApJ, 902, 129
Tröster, T., Ferguson, C., Harnois-Déraps, J., & McCarthy, I. G. 2019, MNRAS, 487, L24
Villaescusa-Navarro, F., Anglés-Alcázar, D., Genel, S., et al. 2021a, ApJ, 915, 71
Villaescusa-Navarro, F., Anglés-Alcázar, D., Genel, S., et al. 2021b, arXiv:2109.09747
Villaescusa-Navarro, F., Genel, S., Angles-Alcazar, D., et al. 2021c, arXiv:2109.10360
Villaescusa-Navarro, F., Genel, S., Castorina, E., et al. 2018, ApJ, 866, 135
Villaescusa-Navarro, F., Wandelt, B. D., Anglés-Alcázar, D., et al. 2020, arXiv:2011.05992
Villanueva-Domingo, P., & Villaescusa-Navarro, F. 2021, ApJ, 907, 44
Villanueva-Domingo, P., Villaescusa-Navarro, F., Anglés-Alcázar, D., et al. 2021a, arXiv:2111.08683
Villanueva-Domingo, P., Villaescusa-Navarro, F., Genel, S., et al. 2021b, arXiv:2111.14874
Wadekar, D., Villaescusa-Navarro, F., Ho, S., & Perreault-Levasseur, L. 2020, arXiv:2012.00111
Wadekar, D., Villaescusa-Navarro, F., Ho, S., & Perreault-Levasseur, L. 2021, ApJ, 916, 42
Weinberger, R., Springel, V., Hernquist, L., et al. 2017, MNRAS, 465, 3291
Yip, J. H. T., Zhang, X., Wang, Y., et al. 2019, arXiv:1910.07813
Zhang, X., Wang, Y., Zhang, W., et al. 2019, arXiv:1902.05965