MNRAS **000**, 1–14 (2022) Preprint 26 May 2022 Compiled using MNRAS LATEX style file v3.0

Machine-guided Exploration and Calibration of Astrophysical Simulations

Boon Kiat Oh,[1]★ Hongjun An,[1,2] Eun-jin Shin,[1] Ji-hoon Kim [1,3]†
and Sungwook E. Hong (홍성욱)[4,5]

[1]*Center for Theoretical Physics, Department of Physics and Astronomy, Seoul National University, Seoul 08826, Korea*
[2]*Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Korea*
[3]*Seoul National University Astronomy Research Center, Seoul 08826, Korea*
[4]*Korea Astronomy and Space Science Institute, 776 Daedeok-daero, Yuseong-gu, Daejeon 34055, Korea*
[5]*Astronomy Campus, University of Science & Technology, 776 Daedeok-daero, Yuseong-gu, Daejeon 34055, Korea*

Accepted XXX. Received YYY; in original form ZZZ

ABSTRACT

We apply a novel method with machine learning to calibrate sub-grid models within numerical simulation codes to achieve convergence with observations and between different codes. It utilizes active learning and neural density estimators. The hyper parameters of the machine are calibrated with a well-defined projectile motion problem. Then, using a set of 22 cosmological zoom simulations, we tune the parameters of a popular star formation and feedback model within `Enzo` to match observations. The parameters that are adjusted include the star formation efficiency, coupling of thermal energy from stellar feedback, and volume into which the energy is deposited. This number translates to a factor of more than three improvements over manual calibration. Despite using fewer simulations, we obtain a better agreement to the observed baryon makeup of a Milky-Way (MW) sized halo. Switching to a different strategy, we improve the consistency of the recommended parameters from the machine. Given the success of the calibration, we then apply the technique to reconcile metal transport between grid-based and particle-based simulation codes using an isolated galaxy. It is an improvement over manual exploration while hinting at a less known relation between the diffusion coefficient and the metal mass in the halo region. The exploration and calibration of the parameters of the sub-grid models with a machine learning approach is concluded to be versatile and directly applicable to different problems.

Key words: galaxies:formation – galaxies:evolution – galaxies:haloes

1 INTRODUCTION

Numerical experiments in cosmology aim to capture the structure formation and evolution in the universe driven by dark matter and baryons. The nonlinear interplay between the two components has been modeled and studied with highly accurate computer simulations. As the computational capability advanced, there has been corresponding improvements in the resolution over the years (Efstathiou et al. 1985; Moore et al. 1999; Springel et al. 2008; Klypin et al. 2011). Despite this improvement, the resolution required to resolve baryonic processes such as star formation, feedback from stars, and black holes in a cosmological setting remains a challenge. Instead, subgrid models are used to represent these processes in simulations (Springel & Hernquist 2003; Governato et al. 2010; Agertz et al. 2013; Shimizu et al. 2019). For the simulations to have predictive power, the parameters of the subgrid models has to be calibrated to reproduce observations (Okamoto et al. 2014; Schaye et al. 2015; Oh et al. 2020).

Feedback stems from the star formation and active galactic nuclei (AGN) activity (Sijacki et al. 2007; Booth & Schaye 2009; Teyssier et al. 2011). It is an important component and can drive the relation between stellar mass and metallicity (De Rossi et al. 2007; Davé et al. 2011). The outflow from the supernova (SN) feedback can also efficiently transports the metal-enriched materials away from the innermost region of the galaxy, regulating the metallicity (Larson 1974; Tremonti et al. 2004; De Rossi et al. 2017). Therefore, feedback plays a pivotal role in recreating realistic galaxies in simulations.

Each form of feedback has a different degree of impact depending on the mass of the halo. To implement them in numerical simulations depends on the various interpretation of these processes. Across different simulation codes, there is a variety of models for SN feedback alone (Cen & Ostriker 2006; Dubois & Teyssier 2008;

★ Corresponding authors: bkoh1986@snu.ac.kr
† Corresponding authors: me@jihoonkim.org

Dalla Vecchia & Schaye 2012; Smith et al. 2018). It can be designed as a thermal injection of energy or mechanical feedback (Kimm et al. 2015; Hopkins et al. 2018a). The latter is found to be relatively resolution-independent and ensures convergence, provided that the resolution of the simulations are high enough.

Given the huge variation in implementation, there is a corresponding difference in the properties of galaxies in simulations (Thacker & Couchman 2000; Springel & Hernquist 2003; Okamoto et al. 2005; Oppenheimer & Davé 2006; Schaye et al. 2010). This diversity is further compounded by the discrepancy in the adopted simulation codes, which can broadly be classified as grid-based or particle-based. Essentially, the difference lies in the way that gas is treated in the simulation, either deposited on a grid or modeled as a particle.

The *AGORA* project was born to investigate both the convergence and divergence of the various simulation codes (Kim et al. 2014, 2016). One of the key findings of Kim et al. (2016) points to the discrepancy in how metal is transported around the simulation domain by different simulation codes. In a grid-based code, metal enrichment of the hot diffuse gas can be found in the halo region due to stronger outflows and non-zero gas density present in the cells. However, a lack of gas in the halo region in particle-based codes, which is represented by the absence of particles leads to metal enrichment confined to the dense star-forming region due to a lack of ram pressure. This difference can be alleviated by the inclusion of an explicit metal diffusion scheme, adding gas particles to the halo region and boosting feedback energy in particle-based codes (Shin et al. 2021). Therefore, feedback plays a pivotal role in simulations. Not only does it determine the baryon makeup within haloes, it provides the means to distribute the metals around the galaxy. Due to the nonlinear nature of how the properties of a simulated galaxy is shaped by the choice of feedback parameters, it is important to explore and understand the impact of the parameter space in these subgrid models.

Amidst all the uncertainty introduced by the different simulation codes and subgrid routines, there exists a need to calibrate certain aspects of the simulation to observations (Okamoto et al. 2014; Schaye et al. 2015). This step is necessary so that the simulation can serve as a testbed for non-calibrated features and possess predictive power. For instance, the 'Evolution and Assembly of GaLaxies and their Environments' (EAGLE) simulation project is calibrated to reproduce several observed properties such as the $z = 0.1$ galaxy stellar mass function (GSMF), the relation between the mass of galaxies and their central black holes and realistic galaxy sizes (Schaye et al. 2015). Oh et al. (2020) calibrated their suite of simulation to reproduce the baryon makeup of a Milky-Way (MW) sized galaxy at $z = 0$, fueled by the observations from McGaugh et al. (2009). These calibrations are costly and prohibit a complete exploration of parameter space. Also, there is minimal scatter associated with the properties presented by McGaugh et al. (2009), which drives the need to apply machine learning approaches in an attempt to match simulated and observed properties to higher precision. Hence, there is a need to streamline and focus the resources on regions of parameter space most likely to yield a desirable outcome. Given the nonlinear nature of the problem, it is difficult to identify the mentioned region, which adds to the motivation of applying machine learning methods.

Machine learning has been gaining a lot of attention as a useful tool that can be applied to a wide range of fields. It has yielded great results in different studies such as pattern recognition (Noh et al. 2015; Sermanet et al. 2013), ill-posed inverse problem (Fessler 2010; Yoon et al. 2018) and properties estimation (Papamakarios et al. 2019; Jo & Kim 2019). It has also been used to generate high resolution simulations from low resolution runs (Li et al. 2021; Ni et al. 2021) and predict cosmological parameters from dark matter particles' positions and velocities in a simulation (Mathuriya et al. 2018). These achievements highlighted an improvement over conventional methods. However, machine learning methods are usually fueled by large amounts of data and the performance is enhanced by the availability of more data. This requirement demands a large amount of computation resources and is time-consuming, which is not feasible for the calibration problem we are interested in.

However, Alsing et al. (2019) proposed a density-estimation likelihood-free inference (DELFI) method, which used a small number of data points by utilizing active learning and neural density estimator. Active learning is one of the machine learning techniques that reduce the demand on the amount of required data (Settles 2009). This technique allows a more effective training through the interaction between the machine and data generation to determine which additional data is needed. This technique is particularly applicable in our case due to the prohibitively expensive nature of simulations.

In this paper, we propose a new and fast method for calibrating the sub-grid models of numerical simulations using machine learning approaches. We apply it to calibrate the baryon makeup of a MW-sized halo at $z = 0$, identical to Oh et al. (2020). We will compare the outcome in terms of the degree of agreement to observations and the amount of resources spent. Combining the results, we will discuss the ability of the model introduced by Cen & Ostriker (1992) and implemented in `Enzo` (Bryan et al. 2014) to match observations. Also, we will apply the method to reconcile the metal transport between a grid-based and a particle-based code as described in Shin et al. (2021). We will probe the versatility of the machine to be applied to different problems. Similar to before, we compare the degree of agreement and resources spent using machine learning methods to manual calibration.

This paper is structured as follows. Section 2 outlines the initial conditions of the simulations, simulation code, and setups to evolve the simulations. Also, we explain the methodology of the machine learning technique. In Section 3, we calibrate the hyper parameters of our machine with a simplified problem: a projectile motion with air resistance. The optimized setup is then applied to the calibration of the star formation and feedback model and to resolve the metal transport discrepancy between grid-based and particle-based code. We will present how the new technique reduces the demand for computational resources and improves the agreement between the simulation and observations, and between different simulation codes. The limitations highlighted by the machine will also be discussed. Lastly, we summarise the paper with discussions and future works in Section 4.

2 METHODOLOGY

2.1 Simulations and their subgrid parameters

2.1.1 Feedback calibration in cosmological simulation

This section provides a summary of the simulation setup which is identical to that presented in Oh et al. (2020) and Kiat Oh et al. (2021). We focus on a MW-sized halo in which the AGN feedback is relatively less significant (Bower et al. 2006; Behroozi et al. 2010; Storchi-Bergmann 2014). We adopt a cosmology consistent with WMAP-9 (Bennett et al. 2013) and the parameters are $\Omega_m = 0.285$, $\Omega_\Lambda = 0.715, \Omega_b = 0.0461, h = 0.695$ and $\sigma_8 = 0.828$ with the usual

definitions. The initial conditions are generated with the MUlti-scale Initial Conditions (`MUSIC`) for cosmological simulations (Hahn & Abel 2011) and the zoom simulations are derived from the parent simulation with a volume of $L = 100\,h^{-1}$ cMpc with 256^3 particles.

We evolve the simulations with an adaptive mesh-refinement (AMR) code (Bryan et al. 2014), `Enzo` and couple the `ZEUS` (Stone & Norman 1992) hydro solver with an N-body adaptive particle-mesh gravity solver (Efstathiou et al. 1985). The chemistry and cooling processes are handled by the `Grackle` library (Smith et al. 2017). In particular, we employ the equilibrium cooling mode from `Grackle`, which utilizes the tabulated cooling rates derived from the photoionization code CLOUDY (Ferland et al. 2013) together with the heating from UV background radiation given by Haardt & Madau (2012).

We will focus on Setup 2 adopted in Oh et al. (2020). The calibration is based on the modified star formation and feedback model described in Cen & Ostriker (1992), looking primarily at f_*, ϵ and r_s. These parameters can be broadly interpreted as star formation efficiency, a proxy for feedback energy budget and extent of feedback injection. We will define these parameters in greater details when they are mentioned in the star formation and feedback processes that follows. They will influence the baryon makeup in the halo at $z = 0$. With three nested levels and five additional levels of AMR, the simulation has a dark matter particle mass resolution of $1.104 \times 10^7\,M_\odot$ and a maximum spatial resolution of 2.196 comoving kpc (ckpc).

There are several conditions that deem a cloud of gas capable of forming a star:

(i) No further refinement within the cell
(ii) Gas density greater than a threshold density: $\rho_{\rm gas} > \rho_{\rm threshold}$
(iii) Convergent flow: $\boldsymbol{\nabla} \cdot \mathbf{v} < 0$
(iv) Cooling time less than a dynamical time: $t_{\rm cool} < t_{\rm dyn}$
(v) Gas mass larger than the Jeans mass: $m_{\rm gas} > m_{\rm jeans}$
(vi) Star particle mass is greater than a threshold mass

Once these conditions are fulfilled, a star particle will be inserted with a mass of

$$m_* = m_{\rm gas} \times f_*, \tag{1}$$

where $m_{\rm gas}$ is the gas mass in the cell and f_* is the parameter governing the conversion efficiency of the gas mass within a cell into a star particle. It is placed in the center of the cell with the same peculiar velocity as the gas in the cell. An equivalent mass of gas to that of the star particle is then removed from the cell to ensure mass conservation.

Despite the instantaneous creation of the star particle, the feedback happens over a longer period of time, to mimic the gradual process of star formation. Within each timestep (dt), the amount of star-forming gas is given by

$$m_{\rm form} = m_* \left[\left(1 + \frac{t - t_0}{t_{\rm dyn}}\right) \exp\left(-\frac{t - t_0}{t_{\rm dyn}}\right) - \left(1 + \frac{t + dt - t_0}{t_{\rm dyn}}\right) \exp\left(-\frac{t + dt - t_0}{t_{\rm dyn}}\right) \right], \tag{2}$$

where t_0 and t are the creation time of the star particle and current time in the simulation respectively. Through this implementation, according to Equation 2, the rate of star formation increases linearly and peaks after one dynamical time before declining exponentially (Smith et al. 2011). The feedback injects mass, energy, and metals per timestep and continues until 12 dynamical times after its creation.

Table 1. Conversion between r_s and feedback volume

r_s	feedback volume [ckpc^3]
1_1	63.50
1_2	201.08
1_3	285.74
2_6	1322.87

As we are only interested in calibrating the amount of feedback energy injected, we will focus on the following to describe how it is being calculated. Interested readers can refer to Section 2.2 in Oh et al. (2020) for the discussion on mass and metals. The amount of feedback energy injected is regulated by ϵ

$$E_{\rm feedback} = m_{\rm form} \times c^2 \times \epsilon \tag{3}$$

where ϵ and c are the feedback efficiency and speed of light respectively. As a reference, for $\epsilon = 10^{-5}$ (Cen & Ostriker 1992), 10^{51} erg of energy is injected per approximately $56\,{\rm M}_\odot$ of stars formed.

Lastly, r_s allows the user to define the volume into which the feedback energy is injected into. For example, r_s = 1_1 refers to the six adjacent cells to the star particle. Increasing this parameter to 1_2 or 1_3 increases the number of cells and hence volume. However, it decreases the amount of feedback energy per cell (refer to Section 2.2.2 in Oh et al. (2020) for more details). To put it simply, r_s = 1_1 $\approx 63.5\,{\rm ckpc}^3$ and r_s = 1_3 $\approx 285.7\,{\rm ckpc}^3$. Refer to Table 1 for the conversion between the notation and the corresponding volume.

We are calibrating f_*, ϵ and r_s to the baryon makeup, specifically the ratio of stars and gas to the total baryonic mass of a MW-sized halo at $z = 0$. The ratios are quantified in the same way to that presented in McGaugh et al. (2010). Using different methods to obtain the equivalent circular velocity, the authors determined the total mass budgets in a variety of galaxies. They used r_{500}, the radius which encloses a density 500 times the critical density of the universe, according to the cluster data used in the analysis. Identical to McGaugh et al. (2010), we quantify the MW-sized halo with the ratio of expected baryons that are detected,

$$f_{\rm d} = \frac{m_{\rm b}}{f_{\rm b} \times m_{500}}, \tag{4}$$

and the ratio of stars,

$$f_{\rm s} = \frac{m_*}{f_{\rm b} \times m_{500}}, \tag{5}$$

where $m_{\rm b}$ and m_{500} refer to the baryonic and total mass within r_{500} respectively, and f_b is the universal baryon fraction determined to be 0.17 ± 0.01 (Komatsu et al. 2009).

We then compare these values with that obtained from the analytical fit to McGaugh et al. (2010), given by equations 10 to 13 and Figure 2 of Oh et al. (2020). The difference between the simulated and observed properties is quantified by $|\Delta f_{\rm d}|$ and $|\Delta f_{\rm s}|$. To obtain a perfect match between the simulated and observed MW-sized galaxy, $\Delta f_{\rm d}$ and $\Delta f_{\rm s}$ should be zero.

The simulated galaxy is also compared to the Kennicutt-Schmidt (KS) relation. It provides a tight correlation between gas surface density and star formation rate (SFR) per unit area Schmidt (1959); Kennicutt (1989, 1998); Kennicutt et al. (2007); Bigiel et al. (2008). A similar comparison was attempted in Oh et al. (2020).

The SFRs are obtained from the stellar mass and averaged over the past 50 Myrs of the simulation at $z = 0$. They are then deposited onto a fixed resolution grid of 750 pc (Bigiel et al. 2008). It will be compared to both results from Bigiel et al. (2008) and

$$\log \Sigma_{\rm SFR} = 1.37 \log \Sigma_{\rm gas} - 3.78, \tag{6}$$

from the best observational fit given by Equation 8 in Kennicutt et al. (2007) where $\Sigma_{\rm SFR}$ and $\Sigma_{\rm gas}$ are expressed in units of $M_\odot\,{\rm yr}^{-1}\,{\rm kpc}^{-2}$ and $M_\odot\,{\rm kpc}^{-2}$ respectively.

2.1.2 *Metallicity distribution simulation*

The aim of this part of the study is to use machine learning methods to calibrate the parameters required to obtain an agreement between a particle-based code (`GIZMO`; Hopkins 2015) and a grid-based code (`Enzo`) at a level consistent with that in Shin et al. (2021). We initialize an ideal simulation of an isolated Milky Way-sized disk galaxy with the initial conditions from the *AGORA* Project (Kim et al. 2016). It models a dark matter halo following the Navarro-Frenk-White (NFW) profile (Navarro et al. 1997), an exponential disk of stars and gas and a stellar bulge with a Hernquist profile (Hernquist 1990). [1] For detailed properties, readers are directed to Table 1 in Shin et al. (2021). They will be evolved with `Enzo` and `GIZMO` using the pressure-energy smoothed particle hydrodynamics (hereafter PSPH; Hopkins 2013).

Instead of the default setup in *AGORA*, following the findings of Shin et al. (2021), we insert additional gas particles in the halo region with $n_{\rm H} = 10^{-6}\,{\rm cm}^{-3}$ for the initial conditions of the simulation with `GIZMO`. The mass of these particles matches that of those in the disk region, $m_{\rm gas,IC} = 8.593 \times 10^4\,M_\odot$, translating to a total of 4,000 gas particles in the halo. They are given an initial metallicity and temperature of $Z_{\rm halo} = 10^{-6}\,Z_{\rm disk}$ and 10^6 K respectively. This modification is necessary to reconcile the metal transport behavior between a grid-based and a particle-based simulation code.

We apply the cubic spline kernel (Hernquist & Katz 1989) for the softening of the gravitational force with the desired number of neighboring particles $N_{\rm ngb} = 32$. We also adopt the Plummer equivalent gravitational softening length $\epsilon_{\rm grav}$ of 80 pc and allow the hydrodynamic smoothing length to reach a minimum of $0.2\epsilon_{\rm grav}$. Radiative cooling, star formation, and feedback will follow the prescription in Kim et al. (2016).

The `Enzo` simulation is the baseline for comparison to the `GIZMO` simulation. We vary the diffusion coefficient, $C_{\rm d}$, and the thermal feedback energy budget in the `GIZMO` simulation. $C_{\rm d}$ is used in the metal diffusion scheme in Hopkins et al. (2018b) and Escala et al. (2018), based on the Smagorinsky-Lilly model (Smagorinsky 1963; Shen et al. 2010). It determines the effect of the subgrid diffusion influenced by the velocity shear between the particles, with the assumption that the local diffusivity is related to the velocity shear and resolution scale (refer to Section 2.3.3 in Shin et al. (2021) for more details). On the other hand, the amount of thermal feedback energy impacts the extent to which the metals are distributed and transported away from the central region of the galaxy. The halo metal mass ($M_{\rm metal}$) and gas mass of metallicity between $0.00885\,Z_\odot$ and $0.0115\,Z_\odot$ ($M_{\rm gas,Z\sim0.01}$) after 500 Myr of evolution will be compared between the two simulations. We choose to use $M_{\rm gas,Z\sim0.01}$ because the gas within this metallicity range is found to be highly sensitive to $C_{\rm d}$ as illustrated in Figure 9 of Shin et al. (2021). This small range of metallicity is chosen arbitrarily to keep the output required for the training of the machine simple.

[1] The *AGORA* initial conditions are publicly available at `http://sites.google.com/site/santacruzcomparisonproject/blogs/quicklinks/`.

2.2 Machine learning architecture and approach

2.2.1 *Neural density estimator*

Given finite computational resources and time, it is impossible to fully explore the parameter space of the subgrid physics in a numerical simulation as described in the previous sections. However, this step is necessary and vital for the calibration of the parameters in order for the simulation to produce results matching observations. We attempt to circumvent this problem by applying machine learning methods. While most machine learning methods require large data, we decide to utilize a neural density estimator, which makes use of a small number of data by representing the recommendation with a probability distribution. The values of the recommendation are then obtained by randomly sampling the output distribution.

We use a mixture density network (Bishop 1994) as the neural density estimator. This model represents the distribution with a set of Gaussian using the estimated mean, standard deviation, and probability of each Gaussian as an output to the given input. To train the network, we use the cross-entropy loss function

$$\mathcal{L} = -\log \sum_{i=1}^{m} p_i(x) N(y|\mu_i(x), \sigma_i(x)) \tag{7}$$

where x and y is the input and output, respectively. m is the number of Gaussian, N is the probability density function of the Gaussian distribution, μ and σ is the mean and standard deviation of the Gaussian distribution respectively. The aim is to minimize $\mathcal{L}$ while training the neural network by fitting the mixture of Gaussian distribution to maximize the likelihood on the given data.

To obtain the recommendation of the properties, the subgrid parameters in different studies serve as input to the neural network, giving a mixture of Gaussian distributions and their respective probability. One of the proposed distributions is selected via Gumbel sampling (Jang et al. 2016; Maddison et al. 2016) with the output probability. The selected Gaussian distribution is then randomly sampled to obtain the recommendation. As such, the number of Gaussian distributions is one of the hyper parameters of the neural network, which will be discussed in Section 3.1. We also look at the impact of the number of initial data and the number of additional data per iteration on the performance of the machine.

2.2.2 *Active learning*

In this section, we will describe the architecture of the machine used as shown in Figure 1. We apply active learning, which is a method of machine learning designed to reduce data generation through the interaction between data generation and machine. It is an iterative process that focuses attention on regions of input space that produce the desired output. In this study, we utilize active learning to minimize the total number of simulations required for the calibration, either to match observations (see Section 2.1.1) or to reconcile the results from different simulation codes (see Section 2.1.2).

The pipeline consists of a forward and a backward path. Starting with an initial set of data, which includes both the input subgrid parameters and simulated output properties, the neural network is trained to provide a set of recommended subgrid parameters according to Section 2.2.1. The desired output is given to the machine for the generation of the recommendation. For example, in Section 2.1.1, the ideal case is a perfect match between the simulated and observed properties, i.e., $|\Delta f_{\rm d}| = 0$ and $|\Delta f_{\rm s}| = 0$. This pro-

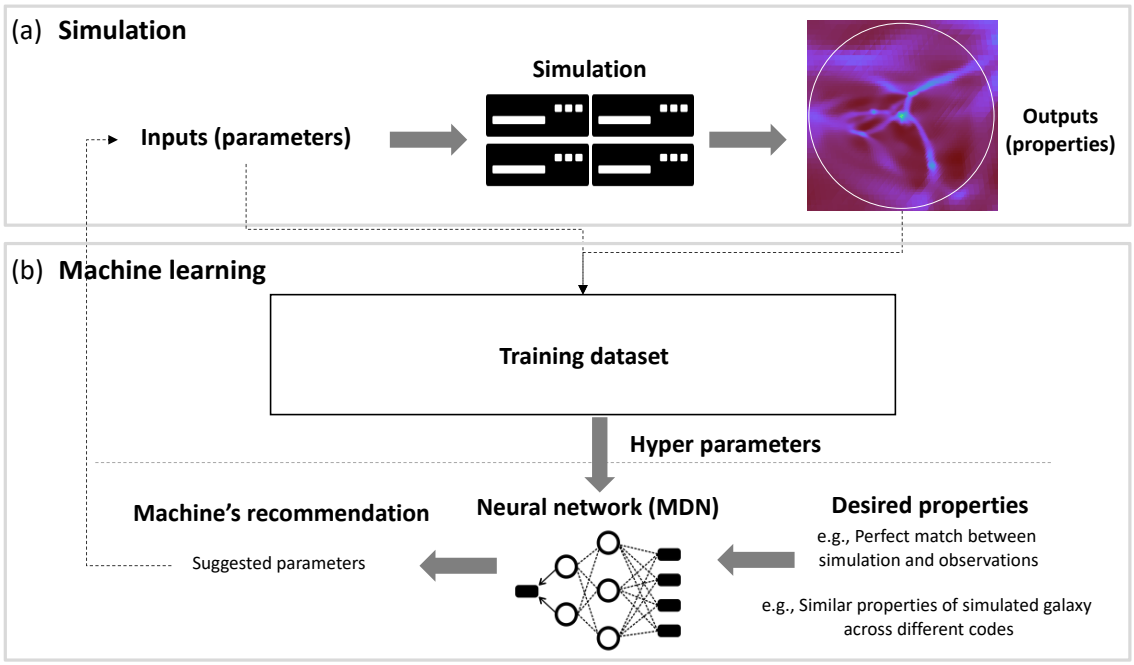

Figure 1. Overview of the calibration using active learning. (a) Simulation part for the forward path. The simulation is performed to obtain parameters from given properties. (b) Machine part for backward path. The results of the simulation are added to the training dataset for the neural network. The trained network then recommends a new set of parameters to match observation. This recommendation is passed to the next forward path. The process terminates when simulated parameters match the desired parameters within a certain limit.

cess constitutes the backward path, in which the machine provides recommended values of subgrid parameters for the simulation.

We then proceed with the forward path. A simulation is carried out with this set of proposed parameters until its desired endpoint. The simulated properties are then noted and checked against the criteria described in the previous section in order to define them as a good match. In the case that the simulation is not, both the input parameters and the output properties are consolidated into a training dataset. Having an additional data point than before, the training data is fed into the backward path again to produce another set of recommendations. This process is repeated until we fulfill the criteria of a good match. For the calibration of feedback parameters in cosmological simulation, a good match is obtained when $|\Delta f_{\rm d}| < 0.0577$ and $|\Delta f_{\rm s}| < 0.0748$ at $z = 0$, given by Oh et al. (2020). On the other hand, for the metal distribution simulations, if $M_{\rm metal}$ and $M_{\rm gas,Z\sim0.01}$ from the `GIZMO` simulation falls within 10% of $1.09 \times 10^6\ M_\odot$ and $2.04 \times 10^6\ M_\odot$ respectively after 500 Myr, it will be classified as a good match. These values of comparison are obtained from an `Enzo` simulation.

The weights of the neural network can be adjusted in two different ways. The first is a reset after every iteration through the forward and backward path while the other is an adjustment of the weights via each iteration. Due to the unique nature of the problem, which is the limited number of data and the high costs of adding every additional data point, we have to prevent a situation of getting stuck in a local minimum as a result of the low number of data. That is why we choose to flush the weights after every iteration.

In the following sections, we will discuss the determination of the hyper parameters of the machine using a projectile motion problem. It is structured to mimic the number of inputs and outputs of the problem described in Section 2.1.1, which is three and two respectively. We can get an estimate of the suitable values by testing this problem with a well-defined solution.

3 RESULTS AND DISCUSSION

3.1 Calibration of hyper parameters using projectile motion with air resistance

In this section, we explore a projectile motion under the influence of air resistance in order to select the hyper parameters of the machine. Ideally, the parameters should be calibrated with a large amount of data that is identical to the problem. However, in a realistic feedback calibration, it is prohibitively expensive to run numerous simulations. Hence, we apply the machine to a well-posed problem: projectile motion with air resistance, in order to explore the hyper parameter space. We set up the machine to possess the same number of inputs and outputs for the neural network to the calibration of feedback parameters in Section 2.1.1. The maximum vertical height and horizontal distance of the projectile motion (two outputs) are calculated from the initial velocity, drag coefficient, and launch angle of the object (three inputs). Using the target of a maximum distance and height of 2.3 m and 0.6 m respectively, we ask the machine to provide recommendations of the inputs required for this projectile motion. With this setup, we tune the number of initial training data, the number of additional data per iteration, and the number of Gaussian distributions on the proximity of the outputs from the recommended inputs to the target.

The calculation is performed using the Euler forward method for the differential equation with a maximum of 100 iterations. The air resistance or Stokes drag applies for very low speed in air and it is linearly increasing with velocity,

$$\ddot{\vec{x}} = -b\dot{\vec{x}} + \vec{g} \tag{8}$$

where $\vec{x}$ denotes a position vector, $\cdot$ denotes derivative with respect to time, b denotes the drag coefficient, and $\vec{g}$ is gravitational acceleration vector. Stokes drag is included to increase the complexity of the problem. The maximum height and distance reached by the projectile motion according to the inputs are then recorded. We allow b, the initial velocity and angle of the projectile to be randomly picked between 0 and 1, 0 m/s to 10 m/s and 0° to 90°, respectively. The recommended velocity and b must be more than zero for the problem to be realistic. It is noted that multiple combinations will produce the desired projectile motion but an example set of inputs to achieve a distance of 2.3 m below a height of 0.6 m is a launch velocity of 5 m/s at an angle of 45 ° with $b = 0.25$.

For the initial training set, we vary the number of data, using intervals of [5, 10, 20, 30, 40, 50]. The number of Gaussian distributions used by the machine and the number of additional data introduced per iteration are fixed to 10 and 1 correspondingly. We then experimented with the number of Gaussian distributions from 5 to 50 in intervals of 5 while fixing the number of initial data to 5 and the number of additional data per iteration to 1. Lastly, we increase the number of additional data per iteration from 1 to 2 while keeping the number of Gaussian distribution and initial data to 20 and 5 respectively. This range of parameters is considered small as compared to typical machine learning problems because it is intended to replicate the situation faced when calibrating a simulation.

The training is performed for 2,000 epochs using the ADAM optimizer (Kingma & Ba 2014) with a learning rate scheduler (ReduceLROnPlateau; initial learning rate: $1e^{-3}$, decay factor: 0.5, patience: 500 epochs, and relative threshold: $1e^{-3}$). To reduce the effects of randomness, the experiment of each comparison is repeated 50 times and the weights initialization is performed with a fixed random seed. The initial data configuration is also fixed. Python and Pytorch library (Paszke et al. 2019) is used for programming and an Nvidia TITAN Xp GPU (Nvidia Corp., Santa Clara, CA) is utilized.

We analyze the exploration of the hyper parameters required to reach within 5% of the total error ($\Delta_{\rm total}$). It is calculated using the target distance $\left(\mathrm{dist}_{\rm target} = 2.3\,\mathrm{m}\right)$ and target height $\left(h_{\rm target} = 0.6\,\mathrm{m}\right)$: $\sqrt{\left(\Delta h/h_{\rm target}\right)^2 + \left(\left(\Delta\mathrm{dist}/\mathrm{dist}_{\rm target}\right)^2\right.}$ where Δh and Δdist is the difference between the height and distance of each throw and their respective target. This optimisation of hyper parameters is shown in Figure 2. We find that using five initial throws yield the smallest amount of additional throws required to reach within 5% error (21.9 ± 20.3). The total number of throws increases proportionally to the number of initial throws. On the other hand, the number of throws required by varying the number of Gaussian distributions does not show a clear tendency as indicated in the second row of Figure 2. Among them, using 20 Gaussian distributions requires the least amount of additional throws to reach within 5% error (14.7 ± 5.1). Lastly, we vary the number of additional throws per iteration. While it is easy to increase the training data by a large amount, we have to remember that each throw is equivalent to a simulation is the problem we are interested in. Hence, the increment is limited to one or two per iteration, shown in third row of Figure 2. We find that increasing the training set by one data per iteration yielded a slightly better performance (14.7 ± 5.1) than two data per iteration (18.2 ± 6.4).

With this setup of 5 initial throws for training, 20 Gaussian distributions for the machine, and one additional throw per iteration, we run the machine for 100 times with different initial training sets to gauge its performance. Figure 3 shows the evolution of the

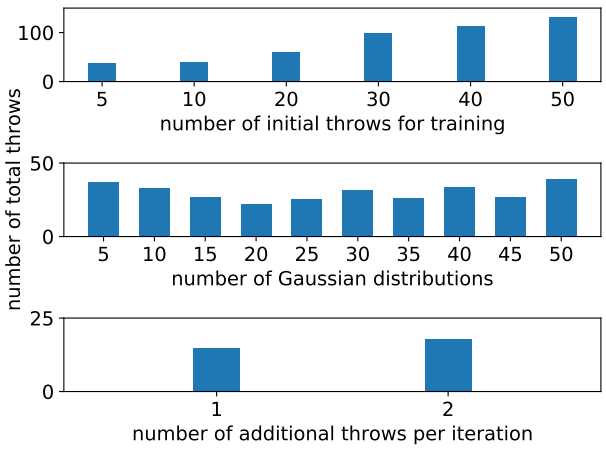

Figure 2. Optimisation of the hyper parameters used in the machine. The top, middle and bottom panel corresponds to the total number of throws required to obtain $\Delta_{\rm total} \leq 5\%$ in relation to the number of initial throws used for training, the number of Gaussian distributions used by the machine and number of additional throws per iteration respectively. Only one parameter is allowed to vary per experiment. We find that five initial throws, 20 Gaussian distributions and one additional throws per iteration yields the minimum number of total throws.

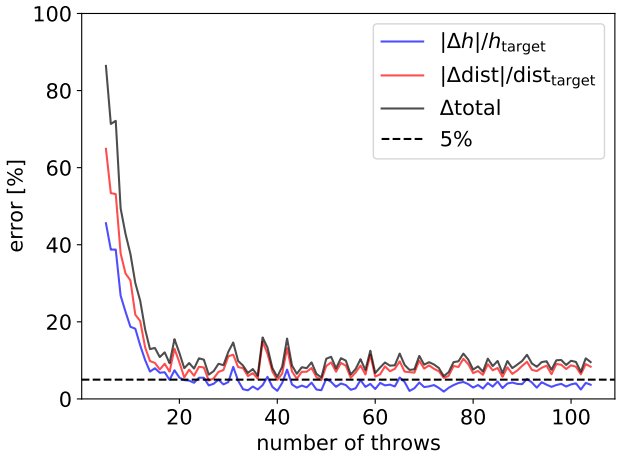

Figure 3. Evolution of the mean Δh (blue), Δdist (red) and Δtotal (black) with the total number of throws in 100 experiments. The dotted line represents the target error range of 5%. In general, the error of all three quantities decreases as the number of throws increases, which indicates the success of the machine to provide meaningful recommendations.

mean error in the distance, height and combination of both with the number of throws. It is evident that as the number of throws increases, the deviations from the target decreases. In other words, the recommendation from the machine is improving the accuracy with each additional throw. Also, the throw reaches within 5% of the target height quicker than the distance, perhaps due to the relatively simpler equation of motion in the y-axis of the throw.

After calibrating the hyper parameters, the machine is then applied to the astrophysical simulations with different strategies of obtaining the recommendation. We will illustrate the discrepancy in the machine's performance with a full exploration (see Section 3.2), explore versus exploit (see Section 3.3), human intervention strategy (see Section 3.5) and modifying the hyper parameters (see

Table 2. Initial set of training data for calibrating the baryon makeup in a MW-sized halo with a Enzo simulation. This table includes the star formation efficiency (f_*), volume in which the feedback is injected into (r_s), feedback efficiency (ϵ), which are the inputs to the machine. The difference between the resulting and observed f_d and f_s (see equations 4 and 5) are the outputs of the machine. It is designed to provide recommendations based on the minimisation of $|\Delta f_d|$ and $|\Delta f_s|$. Refer to Section 3.2 for more information.

Enzo training data				
Inputs (paramters)			Outputs (properties)	
f_*	r_s	ϵ	$\lvert\Delta f_d\rvert$	$\lvert\Delta f_s\rvert$
5.0e-05	2_6	0.1	0.31	0.11
5.0e-06	2_6	0.2	1.11	1.05
2.5e-04	1_3	0.6	0.06	0.18
1.0e-06	1_3	0.9	1.18	1.13
2.0e-05	1_1	0.9	0.26	0.04
Values needed for a perfect match between simulation and observations				
-	-	-	0	0

Section 3.6. These strategies will result in different behaviour and performance of the machine. We can see some indications from the behaviour of the Δtotal as it does not continue to reduce. It means that there is no further exploration of the parameter space around the optimal. It will be discussed and addressed in later sections.

3.2 Calibration of a star formation and feedback model: explore

Together with the machine setup described at the end of the previous section, we apply it to calibrate the star formation and feedback model described in Section 2.1.1. We tune the f_*, ϵ, and r_s to produce the observed f_d and f_s of a MW-sized halo at $z = 0$. The initial training data, which is designed to maximize the coverage in parameter space is shown in Table 2.

Instead of the raw f_d and f_s, we use $|\Delta f_d|$ and $|\Delta f_s|$ (see Section 2.1.1) as the input of the neural network, which means the desired value is zero in both cases. We also represent the r_s with the total volume into which the feedback energy is injected. Together with f_*, we scale both parameters logarithmically to capture slight changes in the values for the output of the neural network. It is important to note that we did not specify that the volume is discrete but the machine picks it up and starts recommending numbers around the discrete values of volume as the number of data points increases. By posing restrictions in the data provided to the machine in each iteration, the machine is able to adapt the recommendations accordingly. With 17 additional simulations, we identify a combination of parameters of (2.7e-5, 0.85, 1_1) which simulated a MW-sized halo with $|\Delta f_d| = 0.00642$ and $|\Delta f_s| = 0.0636$ at $z = 0$.

Due to the exploratory nature of the machine, we can clearly identify the huge variations in the parameters recommended in the left panel of Figure 4. While originally designed to prevent getting trapped in a local minimum, the machine tends to move around the parameter space according to the Gaussian distributions recommended. In other words, unless it is a single distribution, the recommendation will not be consistent. This variation can be seen before and after the yellow region. Right before the best combination, the value of ϵ is still varying drastically. Even the feedback volume is alternating between two values before the best match is found. In contrast, Oh et al. (2020) narrows the variation down to a single parameter in the last steps. After the best match is found, the machine continues to fluctuate around the parameter space, which is not desirable. The machine should focus on these values and attempt to find a better match.

We get a different perspective on the recommendation of the parameters provided by the machine, in the output space (see right panel of Figure 4). As it is a direct comparison to the work by Oh et al. (2020), we use their best-performing simulations to evaluating how well the machine performed. In the mentioned work, the authors found the best match with (3.0e-5, 0.9, 1_1) and (2.5e-5, 0.9, 1_1) which gave a $|\Delta f_d|$ and $|\Delta f_s|$ of 0.0748 and 0.0126 (blue line), and 0.0577 and 0.0447 (green line) respectively. These values are used as constraints under which the properties from our simulations must fall under. The $|\Delta f_d|$ and $|\Delta f_s|$ of our simulated halo are 0.00642 and 0.0636. While the $|\Delta f_s|$ is comparable, $|\Delta f_d|$ is an order of magnitude better than previous results, which means the composition of gas and stars is more accurate. However, the machine fails to continue exploring this region of parameter space (see left panel of Figure 4) and starts suggesting simulations with properties that deviate more from observations. Given the identification of a set of parameters able to reproduce observations, it is desirable that the machine explore this region of parameter space further to possibly locate a better match. For example, if a set of parameters yields properties just above the criteria of acceptance, further exploration around these parameters will be more efficient than trying a completely different region in parameter space.

Within 22 simulations, more than three times fewer simulations, we obtain a set of parameters that produced a MW-sized halo with a baryon makeup that is closer to observations than Oh et al. (2020). This saving translates to an equal amount of computational resources. While acceptable due to our intent to fully explore the parameter space, the recommendations leading up to the best one are random. This behavior is even more undesirable after the best match is found because we hope for the machine to look at this region in greater detail to potentially find a better match.

Though we obtained a better agreement between the simulated and observed properties of the MW-sized halo, it is still not possible to achieve a perfect match, i.e., $|\Delta f_d| = 0$ and $|\Delta f_s| = 0$. Given the inability to do so, we can consider increasing the number of parameters used to calibrate the Cen & Ostriker model of star formation and feedback. Alternatively, there might be inherent limitations in the application of this model, at least within the resolution limits of our simulation. We aim to address the latter by changing the strategy used by the machine to explore the parameter space in detail.

3.3 Calibration of a star formation and feedback model: explore & exploit

The calibration of feedback parameters with the novel method is successful in terms of the agreement and computational resources used but there is huge uncertainty in the exploration of the parameter space. This behavior is attributed to the exploratory nature of the strategy used by the machine. Out of all possible Gaussian distributions identified by the machine, the final set of parameters is drawn randomly within one of the distributions. Rather than ignoring the probability of each Gaussian distribution, we rank them accordingly. It focuses on drawing the recommendations from the distribution with the highest probability if it is higher than the next probable distribution by at least 20%. On top of adjusting the strategy, we also change the set of initial training data to test the robustness of the machine. The five data points are picked to maximize coverage of parameter space, similar to that in Section 3.2. We present them in Table 3.

Adopting this new strategy allows the machine to sample the region that produced the best combination of parameters more thoroughly. While we assume an identical setup of hyperparameters, it

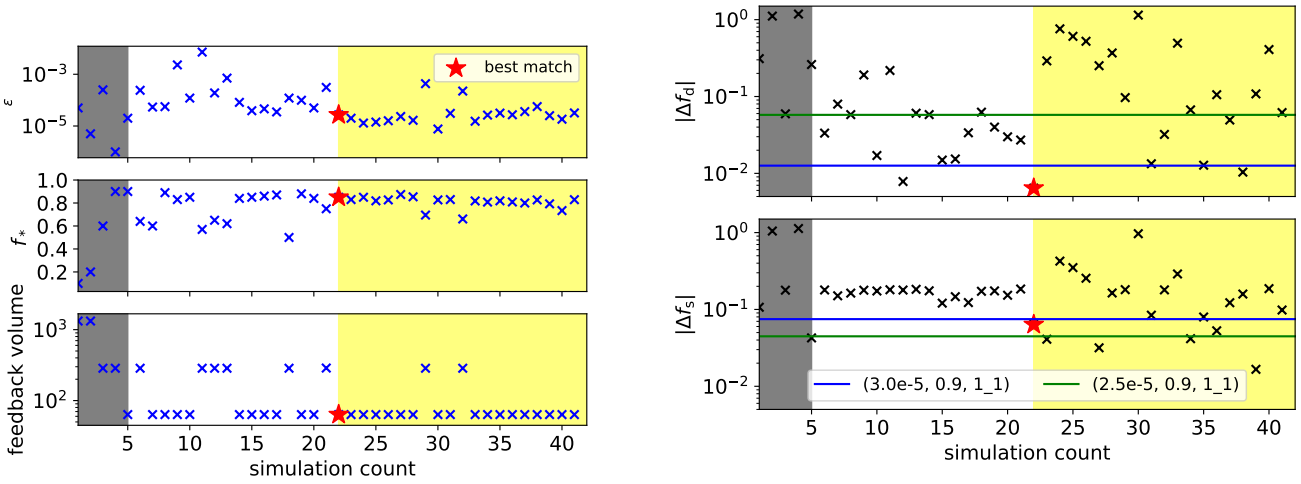

Figure 4. Iteration of simulation parameters recommended by the machine (left) and the corresponding outputs from the simulations (right) using the explore strategy. The blue cross in the left figure represents the ϵ, f_* and r_s in their respective panel. The black cross, blue and green lines in the right panel represents the output of each simulation, output of (3.0e-5, 0.9, 1_1) and (2.5e-5, 0.9, 1_1) from Oh et al. (2020) respectively. The top and bottom rows in the right panel corresponds to the evolution of $|\Delta f_d|$ and $|\Delta f_s|$. In both figures, we have indicated the initial training set consists of the first five data points within the black shaded area and the yellow shaded region illustrates the behavior of the outputs from the recommendations of the machine after the best match (red star) is found. A good match (2.7e-5, 0.85, 1_1) is identified after 17 additional simulations, which is more than a factor of three improvements as compared to Oh et al. (2020). It also yields a $|\Delta f_d|$ which is better than previous work while retaining an acceptable $|\Delta f_s|$, an in-between of those in Oh et al. (2020). We will discuss the progression of the recommendations in Section 3.2.

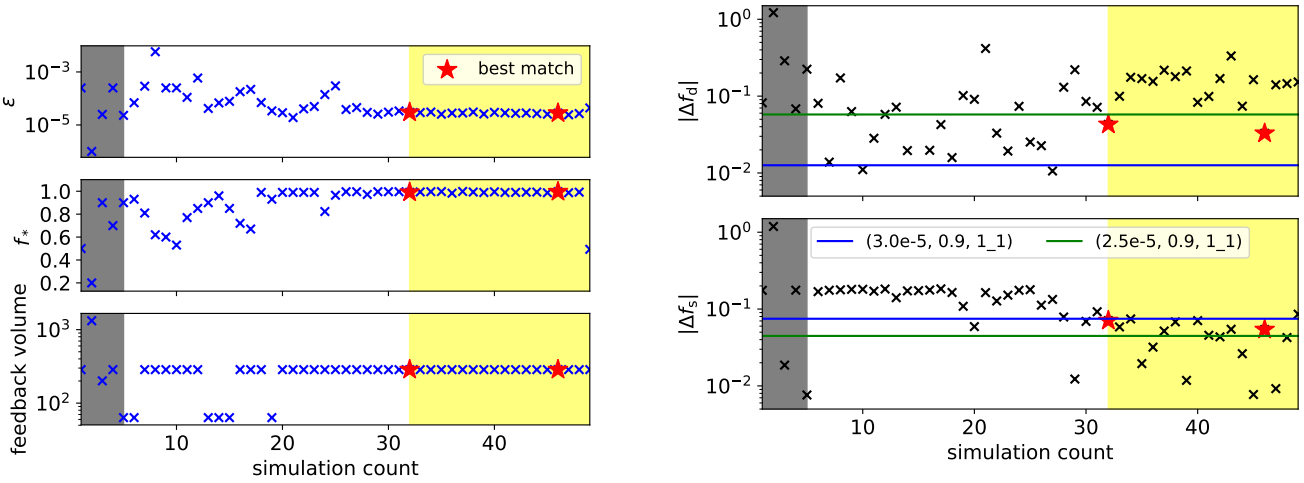

Figure 5. Iteration of simulation parameters recommended by the machine (left) and the corresponding outputs from the simulations (right) using the explore and exploit strategy. The blue cross in the left figure represents the ϵ, f_* and r_s in their respective panel. The black cross, blue and green lines in the right panel represents the output of each simulation, output of (3.0e-5, 0.9, 1_1) and (2.5e-5, 0.9, 1_1) from Oh et al. (2020) respectively. The top and bottom rows in the right panel corresponds to the evolution of $|\Delta f_d|$ and $|\Delta f_s|$. In both figures, we have indicated the initial training set consists of the first five data points within the black shaded area and the yellow shaded region illustrates the behavior of the outputs from the recommendations of the machine after the best match (red star) is found. The best match is given by the machine by 32 simulations and a better one is found by 46 simulations. However, it takes more simulations to reach a worse agreement than Section 3.1. We discuss this discrepancy in Section 3.3.

is worthwhile to investigate it in a future work. As seen from the yellow region in the left panel of figures 4 and 5, the variation in the parameter space decreased significantly in the latter. This difference suggests that by taking the probability into consideration, we can constrain the amount of randomness in the exploration of the parameter space. We find the values capable of reproducing observations to be $f_* = 0.991$, $\epsilon = 3.0 \times 10^{-5}$ and r_s = 1_3, in contrast to Section 3.2 where $f_* = 0.85$, $\epsilon = 2.7 \times 10^{-5}$ and r_s = 1_1. With a higher f_*, ϵ and r_s, our current set of parameters is matching the effect of feedback energy injected per cell in Section 3.2. It is consuming more gas per star particle creation and injecting correspondingly more energy in the increased volume. While the match is not perfect, it creates a MW-sized halo with a baryon makeup as good as Oh et al. (2020).

The other difference is evident after the best set of parameters is determined, where the values stay close to the optimal combination

Table 3. Initial set of training data for calibrating the baryon makeup in a MW-sized halo with a `Enzo` simulation with a different strategy. This table includes the star formation efficiency (f_*), volume in which the feedback is injected into (r_s), feedback efficiency (ϵ) and the difference between the resulting and observed f_s and f_d. The initial training data is different from that in Figure 2 to showcase the robustness of the machine. Refer to Section 3.3 for more information.

`Enzo` training data				
Inputs (parameters)			Outputs (properties)	
f_*	r_s	ϵ	$\lvert\Delta f_d\rvert$	$\lvert\Delta f_s\rvert$
2.5e-04	1_3	0.5	0.08	0.18
1.0e-06	2_6	0.2	1.22	1.19
2.5e-05	1_2	0.9	0.29	0.02
2.5e-04	1_3	0.7	0.07	0.18
2.3e-05	1_1	0.9	0.22	0.01
Values needed for a perfect match between simulation and observations				
-	-	-	0	0

Table 4. Initial set of training data for calibrating the amount of metals in a MW-sized halo with a simulation using `GIZMO` to that using `Enzo`. This table includes the diffusion coefficient (C_d, amount of feedback energy, and the resulting stellar mass, metal mass, and gas mass having a metallicity between 0.00885 $Z_\odot$ and 0.0115 $Z_\odot$. Refer to Section 2.1.2 for more information about the choice of parameters.

`GIZMO` training data			
Inputs (parameters)		Outputs (properties)	
C_d	Feedback energy [$\times 10^{51}$ ergs per SN]	M_{metal} [$\times 10^5\ M_\odot$]	$M_{gas,Z\sim0.01}$ [$\times 10^5\ M_\odot$]
0.006	1	1.73	10.81
0.02	1	1.84	25.07
0.002	1.5	10.61	3.64
0.06	1.5	10.66	30.13
0.02	2	37.86	39.56
Outputs (properties) from `Enzo` simulation			
-	-	10.91	20.40

of parameters as seen in the yellow region of the left panel in Figure 5. We have achieved our goal of better consistency from the recommendation of the machine. However, it takes more simulations to reach an equal or better agreement to Oh et al. (2020) than before. This difference in the number of required simulations can be attributed to this set of parameters being a local rather than a global optimum. We can validate this in the right panel of Figure 5 where $|\Delta f_d|$ and $|\Delta f_s|$ by 32 simulations are more than the best match in the right panel of Figure 4.

Between the right panel of figures 4 and 5, we can see that there are two instead of one set of parameters that managed to simulate a MW-sized halo with a baryon makeup better than Oh et al. (2020). We find a second match in 14 additional simulations (2.8e-5, 0.996, 1_3) after the first (3.0e-5, 0.991, 1_3), giving a decrease in $|\Delta f_d|$ and $|\Delta f_s|$, which is an improvement over the first. For the first match, $|\Delta f_d| = 0.0428$ and $|\Delta f_s| = 0.0708$ while the second match yields $|\Delta f_d| = 0.0328$ and $|\Delta f_s| = 0.0541$. Given the exploiting nature of the strategy in this section, this behavior is as expected at the costs of computational resources.

Despite the input parameters having little variations, the corresponding output properties have significant variations (see Figure 5). Since structure formation and evolution is a highly non-linear process, small variations in the input parameters can have varying degree of impact on the simulation properties at $z = 0$. Furthermore, there is a degree of variance that can be introduced by the simulation code as discussed in Keller et al. (2019) and Oh et al. (2020).

Combining the results of both sections 3.2 and 3.3, we can conclude that the novel application of machine learning methods is successful in improving both the accuracy and computational resources spent on the calibration of the simulations. While Section 3.2 illustrated an effective strategy, it did not allow the possibility of further improving the agreement beyond the initial match. When we then take the possibilities of the Gaussian distributions into account, the machine provides a desirable behavior which is to focus on the parameter space that resulted in a good match. This agreement is however, poorer than before, likely due to the machine focusing on a local minima. If we include human analysis and steer the machine using human intervention, we can potentially further decrease the usage of computational resources. We will explore this strategy and the versatility of the machine in the following sections.

3.4 Calibration of a star formation and feedback model: Kennicutt-Schmidt relation

Similar to Oh et al. (2020), we compare the ability of the simulated galaxies to reproduce the KS relation as described in Section 2.1.1. We contrast the simulations with the fit given by Equation 6 and observations of nearby galaxies from Bigiel et al. (2008) as blue hatched contours. It is shown in Figure 7.

While the simulated galaxy produced properties clustered around the fit, these points do not indicate a distinct slope. Also, the gas density is on the lower end and too low to be compared to observations. This behaviour applies to the best matched simulation regardless of strategy and similar to what Oh et al. (2020) presented.

Given the similarity of the identified parameter space to what Oh et al. (2020) found, it is expected that the agreement to the fit and observation will be similar. The absence of high gas surface density can be explained by the high conversion efficiency of gas into stars, which increases the feedback into the surrounding region. As such, gas is driven out, leaving regions of low gas surface density. This shortcoming can be addressed in future works that will be discussed in Section 4.3.

3.5 Calibration of metal distribution in `GIZMO` simulation with human intervention

To test the robustness of the architecture of the machine described in Section 2.2, we extend its application to a different scientific problem explored in Shin et al. (2021). In this part of the study, we tried to identify values of the diffusion coefficient C_d and feedback energy budget in a `GIZMO` simulation, which will reconcile with the transport behaviour in the `Enzo` simulation. We compare M_{metal} and $M_{gas,Z\sim0.01}$ defined in Section 2.1.2 after evolving for 500 Myr to that obtained in an `Enzo` simulation.

With a similar setup using the hyper parameters described in Section 3.1, a training set comprising of five data sets is provided to the machine. While not comprehensive in the entire parameter space, they are chosen to maximize the coverage of the parameter space in Shin et al. (2021) as seen in Table 4. The training data is illustrated in the black shaded region shown in the left panel of Figure 6.

An identical strategy to Section 3.2 is adopted. The machine is allowed to explore and recommend any values in the parameter space. However, from simulation number six to 14, the machine is oscillating between two prominent Gaussian distributions.

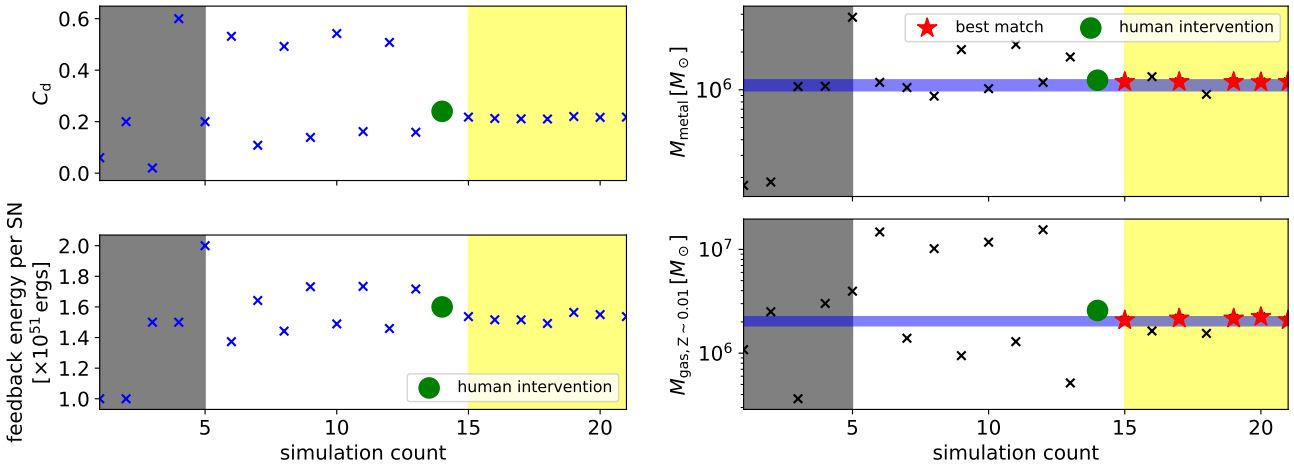

Figure 6. Iteration of simulation parameters recommended by the machine (left) and the corresponding outputs from the simulations (right). The blue cross in the left figure represents the C_d and feedback energy per SN in multiples of 10^{51} ergs. The black cross, blue band, and red star in the right figure represents the output from the simulation, range of target values (±10%) from the `Enzo` simulation, and the simulation that produced values within the blue bands in both panels respectively. In both figures, we have indicated the point of human intervention with the green dot, the initial training set consists of the first five data points within the black shaded area and the yellow shaded region illustrates the behavior of the outputs from the recommendations of the machine after the best match is found. We have decided to have a human-recommended input because of the oscillatory behavior of the parameters prior to the green dot. The details and impact of the human intervention will be discussed in Section 3.5

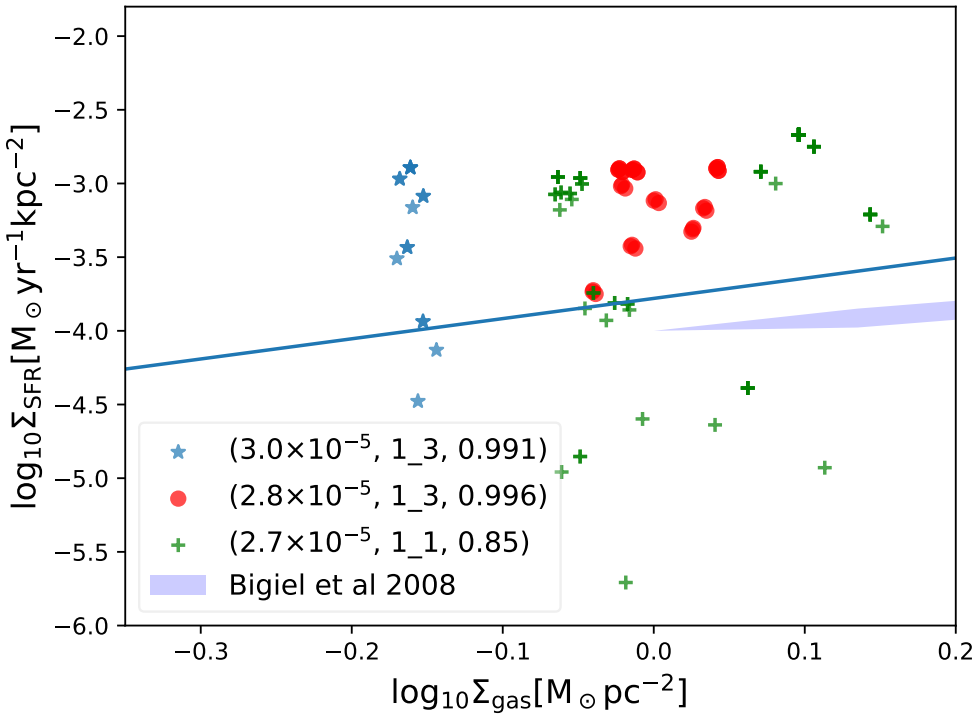

Figure 7. KS relation of SFR surface density against the gas surface density. The different coloured points are simulation data with sub-kpc resolution obtained with various strategies presented in sections 3.2 and 3.3. It includes approximate observations in nearby galaxies by Bigiel et al. (2008) as blue hatched contours and observational fit by Kennicutt et al. (2007) as the blue line. Simulations identified by different strategies produced data points clustered around the fit but possessing low gas surface density. Refer to Section 3.4 for discussion.

It alternates between $C_d \approx 0.05$ with feedback energy per SN $\approx 1.4 \times 10^{51}$ ergs and $C_d \approx 0.014$ with feedback energy per SN $\approx 1.7 \times 10^{51}$ ergs. Since this behaviour comes with a prohibitive cost computationally, we decide to try a new strategy, which is to introduce human intervention. The parameters we have introduced is motivated by the values of the M_{metal} and $M_{gas,Z\sim0.01}$ in the right panel of Figure 6. It is determined to lie between the two oscillating peaks, $C_d = 0.024$ with feedback energy per SN $= 1.6 \times 10^{51}$ ergs.

The initial purpose of the human intervention is to provide the machine with a GIZMO simulation of a MW-sized galaxy that gave properties that resembles closer to that of the Enzo simulation. We tried to steer the machine away from the oscillatory behavior by providing a better match. It worked, evident from the data after the green dot, even resulting in the best match being identified. After running the simulation with the human determined parameters, we obtained a MW-sized halo with M_{metal}, $M_{gas,Z\sim0.01}$ in the GIZMO simulation to within 10% of that in the Enzo simulation.

The recommendations that followed from the best match show consistency in both the parameter and properties space. They do not deviate far from each other or the target values from the Enzo simulation. We use $C_d \approx 0.0218$ and feedback energy per SN $\approx 1.54 \times 10^{51}$ ergs in contrast to $C_d = 0.02$ and feedback energy per SN $\approx 1.8 \times 10^{51}$ ergs in Shin et al. (2021). This discrepancy gives $M_{metal} \approx 1.16 \times 10^6\ M_\odot$ and $M_{gas,Z\sim0.01} \approx 2.08 \times 10^6\ M_\odot$ in our work as compared to $M_{metal} \approx 2.20 \times 10^6\ M_\odot$ and $M_{gas,Z\sim0.01} \approx 2.04 \times 10^6\ M_\odot$ in Shin et al. (2021). While $M_{gas,Z\sim0.01}$ is similar, M_{metal} is significantly different.

The results show that the machine recommended parameters produced a better agreement between the Enzo and GIZMO simulations, showcasing the improvement that is brought about through the usage of the machine learning methods. On top of this increment, the machine highlighted the sensitivity of M_{metal} to C_d. With a higher feedback energy per SN, more metals are lost from the virial radius of the halo due to the rise in metal diffusion caused by the stronger outflow from the disk.

3.6 Calibration of metal distribution in GIZMO simulation by modifying hyper parameters

From the experiment in Section 3.5, we corrected the oscillation of the recommendations obtained from the machine with a human intervention. While it has been proven to be effective in guiding the machine away from the undesirable behaviour, it requires a manual input which might differ from one to the other. In this section, we explore an approach that removes this uncertainty.

Reviewing the architecture of the machine, we recognise that the set of hyper parameters is decided from a projectile motion problem. It is likely that this combination of parameters is not universal for all problems. Hence, we attempt to modify it to prevent the oscillatory behaviour. Specifically, we change the number of Gaussians only because the generation of more training data or more simulation per iteration will place a huge demand on computational resources. It is important to recognise that such an amount of resources might not be available readily.

Hence, we increased the number of Gaussians from 20 to 100 and illustrate the results in Figure 8. The variation of the input parameters and output properties is shown on the left and right panel respectively. In comparison to Figure 6, we can see that the oscillations have been suppressed, indicating that the proposed change is beneficial. After a total of 16 simulations, with $C_d \approx 0.227$ and feedback energy per SN $\approx 1.54 \times 10^{51}$ ergs, we obtain a simulated galaxy with $M_{metal} \approx 1.05 \times 10^6\ M_\odot$ and $M_{gas,Z\sim0.01} \approx 2.15 \times 10^6\ M_\odot$, within 10% of the Enzo simulation.

As discussed, with human intervention, there is a high degree of uncertainty to the choice of values used. Despite both experiments requiring a similar number of simulations to reconcile the properties of the simulated galaxy with a grid based and particle based code, increasing the number of Gaussians is a more deterministic approach. It also points out the inadequacy of the hyper parameters determined in Section 3.1: inability to be applied universally. We will discuss this limitation further in Section 4.2.

4 CONCLUSION AND DISCUSSION

Our study presents the novel technique of calibrating and exploring the subgrid physics parameter space with machine learning methods. It is computationally expensive to run a simulation and it is prohibitively costly if we intend to fully explore the subgrid models. However, it is a necessary and vital step to bridge the results from simulations and observations. We build on the work of Oh et al. (2020) to illustrate the significant savings and increased agreement brought about by the machine learning approach. We summarise our conclusions as follows:

- We use a neural density estimator as the basis of the machine that is used to explore the parameter space. It makes use of a mixture of Gaussian distributions to represent the output. Also, it is suitable for an under-determined problem such as calibrating the subgrid physics model for numerical simulations. It is used in parallel with active learning, allowing interactions between data generation and machine. The aim is to focus the exploration in the region that is of interest determined by the machine. See Section 2.2 for more details.
- The machine consists of a number of hyper parameters that require calibration before it can be applied to numerical simulations. As such, we use a projectile motion problem designed with a similar number of inputs and outputs to the simulation for the determination of the hyper parameters. The inputs are the drag coefficient of the thrown object, velocity, and angle of the throw while the outputs are the maximum height and distance of the resulting projectile motion. We tune the number of Gaussian distributions, the number of initial data as a training set, and the number of additional data per iteration of the machine. The final setup consists of 20 Gaussian distributions, five initial training data, and one additional data per iteration. These values are found to minimize the number of throws to obtain a height and distance within 5% of 2.3 m and 0.6 m respectively. See Section 3.1 for more details.
- We then apply the machine to match the simulation results with observations. As mentioned, the problem is defined with three inputs and two outputs. The inputs are star formation efficiency (f_*), feedback efficiency (ϵ) and feedback volume (r_s) while the outputs are difference in the baryon content ($|\Delta f_d|$) and stellar content ($|\Delta f_s|$) (see Section 2.1.1) between the simulated and observed MW-sized halo at $z = 0$. We obtain a simulated galaxy with $|\Delta f_d| = 0.00642$ and $|\Delta f_s| = 0.0636$ at $z = 0$ using (2.7e-5, 0.85, 1_1). Compared to the results from Oh et al. (2020), we achieve a better agreement to observations and it is obtained with 22 simulations. This number translates to a factor of three improvements over that in Oh et al. (2020). However, due to the exploratory nature of the machine, it did not continue exploring this region of parameter space, which can potentially yield a better match. See Section 3.2 for results and discussion.

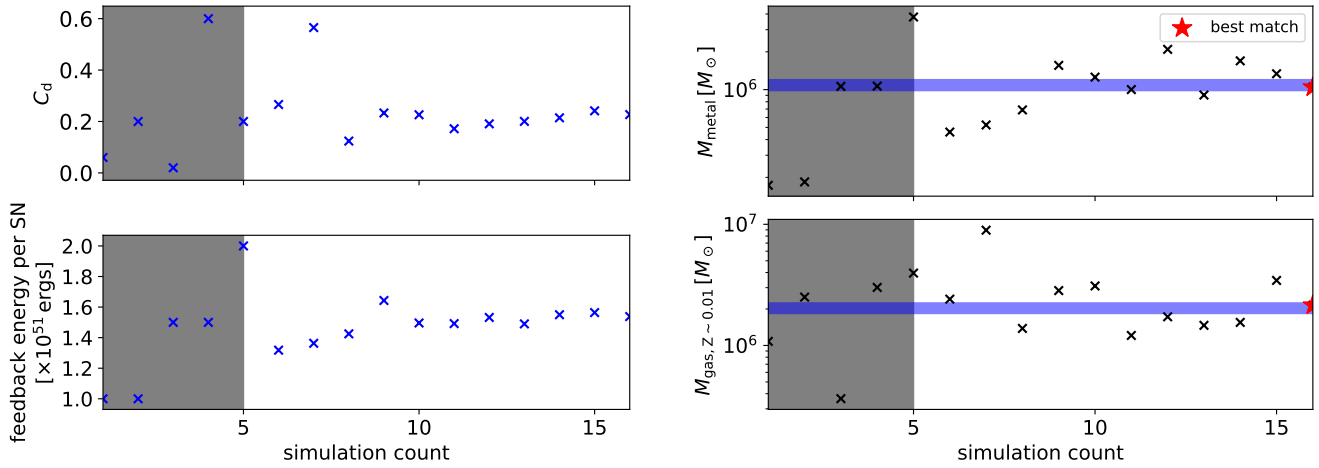

Figure 8. Similar to Figure 6. Instead of introducing a human-recommended input, we increased the number of Gaussians used in the machine learning algorithm to 100. The solution successfully prevented the occurrence of the oscillatory behaviour and pointed out the limitation of the hyper parameters used. These are discussed in Section 3.6

• Since we hope for the machine to continue exploring a particular region of parameter space if deemed able to reproduce observations, we adjusted the strategy used by the machine. Rather than treating each Gaussian peak equally, if the most probable distribution is more than the next by more than 20%, it is designed to focus on the most probable Gaussian distribution. The recommendation will then be provided from this distribution. The machine identifies a set of parameters in a different region of parameter space (3.0e-5, 0.991, 1_3) that produces a baryon makeup at $z = 0$ that is in between Oh et al. (2020) and that obtained with the previous strategy ($|\Delta f_d| = 0.0428$, $|\Delta f_s| = 0.0708$). However, it takes more simulations than before to achieve this aim. Despite using more computational resources, the machine focuses them in this particular region of parameter space and locates another set of parameters (2.8e-5, 0.996, 1_3) which gives a better agreement than the first set ($|\Delta f_d| = 0.0328$, $|\Delta f_s| = 0.0541$). This behavior suggests that the modification to the strategy of the machine worked as intended. See Section 3.3 for results and discussion.

• We also demonstrate the versatility and robustness of the machine to be applied to various problems without modifications. It is used to reconcile the results between a grid-based (`Enzo`) and particle-based (`GIZMO`) simulation code as shown in Shin et al. (2021). For this problem, the inputs that will be varied are the diffusivity and feedback energy per SN of a `GIZMO` simulated isolated disk galaxy. We compare the outputs, which are the M_{metal} and $M_{gas,Z\sim0.01}$ of the `GIZMO` simulated galaxy to that of `Enzo`. A set of parameters is identified with human intervention to produce outputs within 10% of both simulation codes: $C_d \approx 0.0218$ and feedback energy per SN $\approx 1.54 \times 10^{51}$ ergs. The corresponding outputs of the simulations demonstrate the capability of the machine to produce a better agreement. Also, the results pointed out the sensitivity of the halo metal mass to C_d while questioning the relation of $M_{gas,Z\sim0.01}$ to C_d.

• In order to obtain a more deterministic solution than human intervention to resolve the oscillatory behaviour encountered in Section 3.5, we look to modify the values of the hyper parameters. The number of Gaussians is increased from 20 to 100. We showed that this approach effectively suppressed the oscillation in Section 3.6. However, this finding hints at the need to determine the optimal combination of hyper parameters tailored to each problem.

4.1 Limitations of model

Including this work, there have been two attempts at calibrating the Cen & Ostriker (1992) model of star formation and feedback using f_*, ϵ and r_s to reproduce the baryon makeup of a MW-sized halo. However, a perfect match still eludes us, suggesting the presence of limitations. It can lie with the parameters used, i.e., the number of degrees of freedom is insufficient. There are more tunable parameters available and it seems they are necessary to improve the match. However, this increment in the number of parameters means a corresponding increment in the complexity, which further motivates the use of the machine learning method developed in this paper. Alternatively, there might be intrinsic limitations at this relatively higher resolution than the intended resolution of the model.

4.2 Limitations of hyper parameters

While it might not be generic enough to represent the other problems investigated, the hyper parameters determined by the projectile motion problem serve as a beacon to a region of interest in parameter space. However, as we have demonstrated in Section 3.6, this combination of hyper parameters is not universal. If provided with substantially more computational resources and time, a tailored set of hyper parameters can be potentially determined for specific problems. Despite this restriction, with the current set of hyper parameters, the method is already showcasing improvements in terms of both the requirement of resources and the accuracy of the simulated properties.

When we fix the hyper parameters setup, we demonstrated the difference from the strategies used to obtain recommendations from the machine. The list explored in this work is definitely not exhaustive and can even be used in combination. The entire process of calibrating the subgrid physics model can be fully automated but it appears that with additional analysis conducted by the user, a better understanding of the model can be obtained. With an improved grasp of the applied model, a better set of parameters can be identified

within a shorter amount of time, shifting the focus towards scientific analysis of the results.

4.3 Increasing complexity of problem

We have demonstrated the success of applying the machine learning algorithm to guide the exploration and calibration of simulations discussed in this paper. It resulted in generous savings of computational resources spent on numerical simulations. However, we have not fully exploited the capability of the machine learning algorithm.

Simulations are the bottleneck as compared to the speed at which the machine produces recommendation. It takes a day to complete one simulation in contrast to seconds to obtain the suggested parameters. As such, we can increase the complexity of the problem presented to the machine. While we obtained a good match to the properties presented by McGaugh et al. (2009), the agreement to the KS relation is far from ideal. Hence, we can include the KS relation as an additional constraint to the machine, such that it can now consider multiple observations in order to produce a better matching simulation.

We can also seek to apply this method to other major simulations such as the EAGLE simulation or to the *AGORA* project across various simulation codes. It can verify the savings in computational resources that the proposed method brings about. The complexity of the problem can exploit the ability of machine learning in high dimensional space and possibly provide further reduction in the requirement of computational resources while maintaining or even increasing the accuracy of the simulations.

ACKNOWLEDGEMENTS

Ji-hoon Kim acknowledges support by Samsung Science and Technology Foundation under Project Number SSTF-BA1802-04, and by the POSCO Science Fellowship of POSCO TJ Park Foundation. His work was also supported by the National Institute of Supercomputing and Network/Korea Institute of Science and Technology Information with supercomputing resources including technical support, grants KSC-2020-CRE-0219 and KSC-2021-CRE-0442. Sungwook E. Hong was supported by the project 우주거대구조를 이용한 암흑우주 연구 ("Understanding Dark Universe Using Large Scale Structure of the Universe"), funded by the Ministry of Science. The publicly available `ENZO` and `yt` codes used in this work are the products of collaborative efforts by many independent scientists from numerous institutions around the world. Their commitment to open science has helped make this work possible.

DATA AVAILABILITY

The data underlying this article will be shared on reasonable request to the corresponding author.

REFERENCES

Agertz O., Kravtsov A. V., Leitner S. N., Gnedin N. Y., 2013, ApJ, 770, 25
Alsing J., Charnock T., Feeney S., Wandelt B., 2019, Monthly Notices of the Royal Astronomical Society, 488, 4440
Behroozi P. S., Conroy C., Wechsler R. H., 2010, ApJ, 717, 379
Bennett C. L., et al., 2013, ApJS, 208, 20
Bigiel F., Leroy A., Walter F., Brinks E., de Blok W. J. G., Madore B., Thornley M. D., 2008, AJ, 136, 2846
Bishop C. M., 1994
Booth C. M., Schaye J., 2009, MNRAS, 398, 53
Bower R. G., Benson A. J., Malbon R., Helly J. C., Frenk C. S., Baugh C. M., Cole S., Lacey C. G., 2006, MNRAS, 370, 645
Bryan G. L., et al., 2014, ApJS, 211, 19
Cen R., Ostriker J. P., 1992, ApJ, 399, L113
Cen R., Ostriker J. P., 2006, ApJ, 650, 560
Dalla Vecchia C., Schaye J., 2012, MNRAS, 426, 140
Davé R., Oppenheimer B. D., Finlator K., 2011, MNRAS, 415, 11
De Rossi M. E., Tissera P. B., Scannapieco C., 2007, MNRAS, 374, 323
De Rossi M. E., Bower R. G., Font A. S., Schaye J., Theuns T., 2017, MNRAS, 472, 3354
Dubois Y., Teyssier R., 2008, A&A, 477, 79
Efstathiou G., Davis M., White S. D. M., Frenk C. S., 1985, ApJS, 57, 241
Escala I., et al., 2018, MNRAS, 474, 2194
Ferland G. J., et al., 2013, Rev. Mex. Astron. Astrofis., 49, 137
Fessler J. A., 2010, IEEE signal processing magazine, 27, 81
Governato F., et al., 2010, Nature, 463, 203
Haardt F., Madau P., 2012, ApJ, 746, 125
Hahn O., Abel T., 2011, MNRAS, 415, 2101
Hernquist L., 1990, ApJ, 356, 359
Hernquist L., Katz N., 1989, ApJS, 70, 419
Hopkins P. F., 2013, MNRAS, 428, 2840
Hopkins P. F., 2015, MNRAS, 450, 53
Hopkins P. F., et al., 2018a, MNRAS, 477, 1578
Hopkins P. F., et al., 2018b, MNRAS, 480, 800
Jang E., Gu S., Poole B., 2016, arXiv e-prints, p. arXiv:1611.01144
Jo Y., Kim J.-h., 2019, Monthly Notices of the Royal Astronomical Society, 489, 3565
Keller B. W., Wadsley J. W., Wang L., Kruijssen J. M. D., 2019, MNRAS, 482, 2244
Kennicutt Jr. R. C., 1989, ApJ, 344, 685
Kennicutt Jr. R. C., 1998, ApJ, 498, 541
Kennicutt Jr. R. C., et al., 2007, ApJ, 671, 333
Kiat Oh B., Peacock J. A., Khochfar S., Smith B. D., 2021, arXiv e-prints, p. arXiv:2103.02234
Kim J.-h., et al., 2014, ApJS, 210, 14
Kim J.-h., et al., 2016, ApJ, 833, 202
Kimm T., Cen R., Devriendt J., Dubois Y., Slyz A., 2015, MNRAS, 451, 2900
Kingma D. P., Ba J., 2014, arXiv preprint arXiv:1412.6980
Klypin A. A., Trujillo-Gomez S., Primack J., 2011, ApJ, 740, 102
Komatsu E., et al., 2009, ApJS, 180, 330
Larson R. B., 1974, MNRAS, 166, 585
Li Y., Ni Y., Croft R. A. C., Di Matteo T., Bird S., Feng Y., 2021, Proceedings of the National Academy of Science, 118, 2022038118
Maddison C. J., Mnih A., Whye Teh Y., 2016, arXiv e-prints, p. arXiv:1611.00712
Mathuriya A., et al., 2018, arXiv e-prints, p. arXiv:1808.04728
McGaugh S. S., Schombert J. M., De Blok W., Zagursky M. J., 2009, The Astrophysical Journal Letters, 708, L14
McGaugh S. S., Schombert J. M., de Blok W. J. G., Zagursky M. J., 2010, ApJ, 708, L14
Moore B., Ghigna S., Governato F., Lake G., Quinn T., Stadel J., Tozzi P., 1999, ApJ, 524, L19
Navarro J. F., Frenk C. S., White S. D. M., 1997, ApJ, 490, 493
Ni Y., Li Y., Lachance P., Croft R. A. C., Di Matteo T., Bird S., Feng Y., 2021, MNRAS, 507, 1021
Noh H., Hong S., Han B., 2015, in Proceedings of the IEEE international conference on computer vision. pp 1520–1528
Oh B. K., Smith B. D., Peacock J. A., Khochfar S., 2020, MNRAS, 497, 5203
Okamoto T., Eke V. R., Frenk C. S., Jenkins A., 2005, MNRAS, 363, 1299
Okamoto T., Shimizu I., Yoshida N., 2014, PASJ, 66, 70
Oppenheimer B. D., Davé R., 2006, MNRAS, 373, 1265
Papamakarios G., Sterratt D., Murray I., 2019, in The 22nd International Conference on Artificial Intelligence and Statistics. pp 837–848

Paszke A., et al., 2019, in Advances in neural information processing systems. pp 8026–8037
Schaye J., et al., 2010, MNRAS, 402, 1536
Schaye J., et al., 2015, MNRAS, 446, 521
Schmidt M., 1959, ApJ, 129, 243
Sermanet P., Kavukcuoglu K., Chintala S., LeCun Y., 2013, in Proceedings of the IEEE conference on computer vision and pattern recognition. pp 3626–3633
Settles B., 2009, Computer Sciences Technical Report 1648, Active Learning Literature Survey. University of Wisconsin–Madison
Shen S., Wadsley J., Stinson G., 2010, MNRAS, 407, 1581
Shimizu I., Todoroki K., Yajima H., Nagamine K., 2019, MNRAS, 484, 2632
Shin E.-J., Kim J.-H., Oh B. K., 2021, ApJ, 917, 12
Sijacki D., Springel V., Di Matteo T., Hernquist L., 2007, MNRAS, 380, 877
Smagorinsky J., 1963, Monthly Weather Review, 91, 99
Smith B. D., Hallman E. J., Shull J. M., O'Shea B. W., 2011, ApJ, 731, 6
Smith B. D., et al., 2017, MNRAS, 466, 2217
Smith M. C., Sijacki D., Shen S., 2018, MNRAS, 478, 302
Springel V., Hernquist L., 2003, MNRAS, 339, 289
Springel V., et al., 2008, MNRAS, 391, 1685
Stone J. M., Norman M. L., 1992, ApJS, 80, 753
Storchi-Bergmann T., 2014, in Sjouwerman L. O., Lang C. C., Ott J., eds, IAU Symposium Vol. 303, The Galactic Center: Feeding and Feedback in a Normal Galactic Nucleus. pp 354–363 (arXiv:1401.0032), doi:10.1017/S174392131400091X
Teyssier R., Moore B., Martizzi D., Dubois Y., Mayer L., 2011, MNRAS, 414, 195
Thacker R. J., Couchman H. M. P., 2000, ApJ, 545, 728
Tremonti C. A., et al., 2004, ApJ, 613, 898
Yoon J., et al., 2018, Neuroimage, 179, 199

This paper has been typeset from a TEX/LATEX file prepared by the author.