Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **526,** 2903–2919 (2023) https://doi.org/10.1093/mnras/stad2901
Advance Access publication 2023 September 26

MF-Box: multifidelity and multiscale emulation for the matter power spectrum

Ming-Feng Ho [1]★ Simeon Bird [1]★ Martin A. Fernandez [1]★ and Christian R. Shelton[2]

[1]*Department of Physics & Astronomy, University of California, Riverside, 900 University Ave., Riverside, CA 92521, USA*
[2]*Department of Computer Science & Engineering, University of California, Riverside, 900 University Ave., Riverside, CA 92521, USA*

Accepted 2023 September 20. Received 2023 September 13; in original form 2023 June 15

ABSTRACT

We introduce `MF-Box`, an extended version of `MFEmulator`, designed as a fast surrogate for power spectra, trained using *N*-body simulation suites from various box sizes and particle loads. To demonstrate `MF-Box`'s effectiveness, we design simulation suites that include low-fidelity (LF) suites (L1 and L2) at 256 and 100 Mpc h^{-1}, each with 128^3 particles, and a high-fidelity (HF) suite with 512^3 particles at 256 Mpc h^{-1}, representing a higher particle load compared to the LF suites. `MF-Box` acts as a probabilistic resolution correction function, learning most of the cosmological dependencies from L1 and L2 simulations and rectifying resolution differences with just three HF simulations using a Gaussian process. `MF-Box` successfully emulates power spectra from our HF testing set with a relative error of < 3 per cent up to $k \simeq 7\,h\mathrm{Mpc}^{-1}$ at $z \in [0, 3]$, while maintaining a cost similar to our previous multifidelity approach, which was accurate only up to $z = 1$. The addition of an extra LF node in a smaller box significantly improves emulation accuracy for `MF-Box` at $k > 2\,h\mathrm{Mpc}^{-1}$, increasing it by a factor of 10. We conduct an error analysis of `MF-Box` based on computational budget, providing guidance for optimizing budget allocation per fidelity node. Our proposed `MF-Box` enables future surveys to efficiently combine simulation suites of varying quality, effectively expanding the range of emulation capabilities while ensuring cost efficiency.

Key words: methods: statistical – cosmology: theory.

1 INTRODUCTION

Over the past decade, cosmological large-scale structure surveys have evolved increasingly in resolution and size. As observations probe more non-linear structures with high precision, theoretical predictions must be highly accurate to match the observational errors at corresponding small scales. The only way to achieve such accurate predictions is by running *N*-body simulations. However, including expensive numerical simulations in the cosmological inference will require $\sim 10^6$ likelihood evaluations using simulations, i.e. $\sim 10^6$ numerical simulations in the Markov chain Monte Carlo (MCMC) sampling, making it impractical to use simulations for Bayesian inference directly.

In the development of statistical surrogate modelling, emulators emerged as a Bayesian approach to analyse simulations and perform fast function predictions (Currin et al. 1991; Santner, Williams & Notz 2003; O'Hagan 2006). In cosmology, emulators have been widely used as a fast surrogate model to replace the expensive likelihood evaluations in the MCMC sampling. For example, using surrogate models to replace the Boltzmann code in cosmological inference (Auld et al. 2007; Auld, Bridges & Hobson 2008; Aricò, Angulo & Zennaro 2022; Günther et al. 2022; Nygaard et al. 2022; Spurio Mancini et al. 2022). With a large number of training samples ($\sim \mathcal{O}(10^4 - 10^6)$), these Boltzmann code emulators have successfully improved the speed of the current parameter estimation pipeline. Another approach is using surrogates to replace MCMC to emulate the posterior distribution directly, reducing the overall required number of likelihood evaluations (El Gammal et al. 2022).

Unlike the emulators for Boltzmann codes, likelihood evaluations based on numerical simulations, such as cosmological *N*-body simulations, are more expensive per training sample. Therefore, only a limited number of full-size training simulations ($\sim \mathcal{O}(10^1 - 10^2)$) are computationally available. Emulation based on numerical simulations has been implemented in various cosmological applications: the matter power spectrum (Heitmann et al. 2009, 2014; Lawrence et al. 2017; Euclid Collaboration et al. 2019, 2021), baryonfication simulations (Giri & Schneider 2021; Aricò et al. 2021), arbitrary cosmology (Giblin et al. 2019), *f*(*R*) gravity (Arnold et al. 2022; Harnois-Déraps et al. 2022), weak lensing (Harnois-Déraps, Giblin & Joachimi 2019; Davies et al. 2021; Giblin, Cai & Harnois-Déraps 2023), halo mass function (McClintock et al. 2019; Nishimichi et al. 2019; Bocquet et al. 2020), 21-cm power spectrum (Kern et al. 2017) and global signal (Cohen et al. 2020; Bevins et al. 2021; Bye, Portillo & Fialkov 2022), and Lyman-α forest (Bird et al. 2019; Rogers et al. 2019; Pedersen et al. 2021; Rogers & Peiris 2021a, b; Cabayol-Garcia et al. 2023). All these emulators are self-consistent and can replicate the simulations as surrogate models to accelerate the parameter inference pipeline.

Emulators have also been used in several current surveys. Zürcher et al. (2022) used an emulator on Dark Energy Survey year 3 data for cosmic shear peak statistics. Neveux et al. (2022) used an emulator on SDSS quasars and galaxies. Beyond cosmological

★ E-mail: mho026@ucr.edu (M-FH); sbird@ucr.edu (SB); mfern027@ucr.edu (MAF)

inference, Jo et al. (2023) uses emulation to calibrate the galaxy formation simulations. Kugel et al. (2023) and Salcido et al. (2023) build emulators to quantify the subgrid feedback effects in the hydrodynamical simulations. Emulators have also been used in a wide range of disciplines, for example, exoplanet (Rogers et al. 2021), gravitational wave (Cheung et al. 2021), stellar population synthesis (Alsing et al. 2020), heavy-ion physics (Ji et al. 2021, 2022), astrochemistry (Holdship et al. 2021), and biology (Vernon et al. 2018).

The computational costs of cosmological emulators are rapidly increasing, driven by an increase in both survey accuracy and number of model parameters. Over the past few years, cosmological emulators based on N-body simulations have evolved from five-dimensional cosmology [e.g. wCDM in Coyote Universe (Heitmann et al. 2009)] to higher dimensions, for example, eight-dimensional w_0w_aCDM + $\sum m_\nu$ cosmology in Euclid Collaboration et al. (2021) and Mira-Titan Universe (Lawrence et al. 2017; Moran et al. 2023). The increase in dimensionality means the number of simulations required for training an accurate emulator also needs to increase dramatically. For instance, EuclidEmulator2 requires more than 200 high-resolution simulations with 3000^3 in an eight-dimensional cosmology. Moreover, when the astrophysics effects are not ignorable for cosmological inference (Giri & Schneider 2021; Villaescusa-Navarro et al. 2021; Aricò et al. 2021), more expensive simulations, such as hydrodynamical simulations including baryonic effects, must be used for training realistic emulators. This increase in computational cost poses a challenge for the implementation of emulators in future surveys, making them prohibitively expensive and difficult to adopt unless the efficiency of emulation techniques can be improved.

An efficient approach to reducing the computational cost is building emulators using multifidelity emulation (`MFEmulator`), which allows simulations with different particle loads to be combined (Ho, Bird & Shelton 2022). Fernandez, Ho & Bird (2022) showed that it is possible to construct a realistic emulator using hydrodynamical simulations through the `MFEmulator` technique, emulating Lyman-α forest with sub-per cent test accuracy using only six high-fidelity (HF) simulations. In Fernandez et al. (2022) and Ho et al. (2022), we assumed the particle load is the only fidelity variable. This is a limitation, as simulation volumes also correlate with the accuracy of a simulation: With a constant particle load, larger box sizes enhance accuracy at larger scales but diminish it at smaller scales due to reduced mass resolution. Smaller volumes with the same particle load can capture finer small-scale details, though a minimum box size requirement exists (Heitmann et al. 2010; Schneider et al. 2016). Here, we show that the cost of training an `MFEmulator` can be further reduced by having multiple fidelities which vary both simulation volumes and particle loads.

The multifidelity method we use, based on Kennedy & O'Hagan (2000) and Ho et al. (2022), is just one of many multifidelity techniques. Peherstorfer, Willcox & Gunzburger (2018) surveyed the multifidelity methods in uncertainty quantification, inference, and optimization. A few popular methods include the control variate technique, which has been applied in cosmology in Chartier et al. (2021); Chartier & Wandelt (2022) on reducing the variance of the covariance matrix, and multilevel or multistage MCMC (Christen & Fox 2005; Lykkegaard et al. 2020), which use low-fidelity (LF) models to reduce the number of expensive likelihood evaluations in MCMC. Though multilevel MCMC is a promising method, its practical use requires running thousands of N-body simulations in the sampler, which is not yet applicable to cosmological inference. Another similar method is using deep-learning methods to learn the mapping from low- to high-resolution simulations to directly generate the snapshots of the 'super-resolution' simulations (Kodi Ramanah et al. 2020; Li et al. 2021; Ni et al. 2021). While this method shows promise, it is currently limited to a single cosmology and is not yet suitable for inference.

The statistical and computer science literature already contains work on multifidelity techniques with more than one LF node. Lam, Allaire & Willcox (2015); Poloczek, Wang & Frazier (2017) considered a multi-information source framework, which combines more than one information node to achieve an overall lower variance. In this work, we use a graphical Gaussian process, based on a directed acyclic graph (Ji et al. 2021), to predict HF simulations using LF simulations in two different simulation volumes.

A design using multiple LF nodes can be helpful in several ways. One example, which we will show in this work, is enhancing the resolution at small scales using an additional LF node with a smaller box size. A cosmological simulation has strict volume requirements to ensure that the base mode is linear and to beat cosmic variance. However, it also needs high enough particle load (or spatial resolution) to capture the non-linearities at small scales. `MFEmulator` provides a way to improve small-scale structures using a simulation suite from a lower particle load. Nevertheless, the non-linear information in a lower particle-load simulation is also limited. An economical way to resolve small scales is to run simulations in small boxes to increase the spatial resolution by sacrificing some large-scale information.

Another approach to minimizing the number of training simulations is Bayesian optimization, where a sequential choice of new training simulations is designed to optimize the likelihood function globally. For example, Leclercq (2018), Rogers et al. (2019), and Takhtaganov et al. (2021) implemented Bayesian optimization in the cosmological inference. Similar approaches, such as Pellejero-Ibañez et al. (2020), Boruah et al. (2022), Cole et al. (2022), and Neveux et al. (2022), iteratively train emulators on the high likelihood regions of the parameter space, thus minimizing the overall training samples to achieve accurate posterior distribution. Our multifidelity emulation is a complimentary technique, which can be combined with Bayesian optimization for the lowest computational cost.

This paper presents `MF-Box`, extending our previously developed `MFEmulator` to allow multiple LF nodes in a multifidelity emulator. `MF-Box` uses the multifidelity graphical Gaussian process model (GMGP) (Ji et al. 2021) to emulate HF simulations using LF simulations from two different simulation volumes. A GMGP model is an extension of the traditional KO model (Kennedy & O'Hagan 2000) and non-linear autoregressive Gaussian process (NARGP) model (Perdikaris et al. 2017). The difference is that a GMGP allows multiple nodes in a fidelity while KO or NARGP models assume one node per fidelity. For example, in our case, the LF nodes include separate box sizes with the same particle load, resolving different scales of the Universe.

Our references to LF and HF nodes are based on a relative scale within the context of our multifidelity framework. We do not directly compare these definitions to other matter power spectrum emulators. Our primary goal is to demonstrate the effectiveness of `MF-Box` as a probabilistic resolution correction tool. This allows us to correct the resolution of an LF emulator, approximating higher particle loads using a limited number of HF simulations.

Consequently, the focus of our discussion on emulation error revolves around predicting unseen HF simulations in the test set. This choice is intentional, as it allows us to assess how well `MF-Box` can upscale an LF emulator when predicting HF simulation outputs.

Table 1. LF and HF simulation suites used in our study. The definition of LF and HF nodes is based on a relative scale specific to our approach and is not intended for direct comparison with other matter power spectrum emulators.

Simulation	Box volume	$N_{\rm part}$	Node hour
L1	$(256\ \mathrm{Mpc}\ \ h^{-1})^3$	128^3	∼1.0
L2	$(100\ \mathrm{Mpc}\ \ h^{-1})^3$	128^3	∼1.7
HF	$(256\ \mathrm{Mpc}\ \ h^{-1})^3$	512^3	∼140
Test	$(256\ \mathrm{Mpc}\ \ h^{-1})^3$	512^3	∼140

It is worth highlighting that the framework we present here can be adapted for use with various other summary statistics emulators, accommodating different definitions of LF and HF nodes as needed.

We will also present an analysis of the emulation errors in relation to the computational budget. Previous studies Wendland (2004) and Ji et al. (2021) have demonstrated that Gaussian process emulator errors can be bounded by a power-law function. In this paper, we model the emulation error from `MF-Box` as a power-law function of the number of training simulations and empirically infer the emulator error function from our `MF-Box` results. By utilizing this empirical error function, we can estimate the emulation error associated with a given multifidelity design, as well as determine the optimal budget allocation for each node. This error analysis serves as a useful guide for future development of `MFEmulator` techniques.

In Section 2, we will describe our simulations and experimental design. Section 3 will review the single-fidelity emulator as well as three multifidelity emulation methods, namely AR1, NARGP, and `MF-Box`. Our sampling strategy for selecting input cosmologies for HF simulations will be outlined in Section 4. Empirical inference of the emulation error function will be discussed in Section 5. Section 6 will present the results of `MF-Box`, followed by the conclusion in Section 7.

2 SIMULATIONS

We perform dark matter-only simulations using the open-source MP-Gadget code (Feng et al. 2018),[1] an N-body and smoothed particle hydrodynamical (SPH) simulation code derived from Gadget-3 (Springel & Hernquist 2003) and used to run the ASTRID simulation (Bird et al. 2022; Ni et al. 2022), a large-scale high-resolution cosmological simulation with $250\,\mathrm{Mpc}\ \ h^{-1}$ containing 2×5500^3 particles. The base of MP-Gadget is Gadget-3, but, among other improvements, it has been rewritten to take advantage of shared-memory parallelism and the hierarchical time-stepping from Gadget-4 Springel et al. (2021). Detailed descriptions of the simulation code can be found in Bird et al. (2022).

We start the simulations at $z = 99$ and finish at $z = 0$. The initial linear power spectrum and transfer function are produced by CLASS (Lesgourgues 2011) at $z = 99$ through the Zel'dovich approximation (Zel'Dovich 1970). We assume periodic boundary conditions. We use a Fourier transform-based particle-mesh method on large scales for the gravitational forces and a Barnes–Hut tree (Barnes & Hut 1986) on small scales. Table 1 summarizes the simulation volumes and particle loads used in this paper. We use the same set of LF (L1) and HF pairs as in Ho et al. (2022), with an additional LF node (L2) to demonstrate the emulation using simulations from different box sizes. However, the framework presented in this paper is generalizable to more than two LF nodes. Fig. 1 shows a visual illustration for the dark-matter only simulations used in this paper.

[1] https://github.com/MP-Gadget/MP-Gadget/

Our emulation target is the matter power spectrum, $P(k)$, a summary statistic of the overdensity field. We measure the matter power spectrum with a cloud-in-cell mass assignment. We use the built-in power spectrum estimator from MP-Gadget; the power spectrum is thus generated on a mesh the same size as the simulation's PM grid, which is three times the mean interparticle spacing. The multifidelity emulation framework we introduce here is also applicable to other implementations of power spectrum calculations, such as those generated by NBodyKit (Hand et al. 2018).

Fig.2 shows an example of our emulation target: matter power spectra from different resolutions, where the LF simulations (L1 and L2) have two different box sizes. L1 simulations are in the same box size ($256\,\mathrm{Mpc}\ \ h^{-1}$) as HF simulations (HF) with the same initial condition seeding; whereas, L2 simulations have a smaller box size ($100\,\mathrm{Mpc}\ \ h^{-1}$) than L1 and HF. In principle, L2 can capture more small-scale structures due to its smaller box size. Indeed, as shown in the second, third, and bottom panels in Fig. 2, L2 is more accurate than L1 at small scales. For example, at $z = 3$, L2/HF is closer to 1 than L1/HF at small scales ($k > 0.6\,h\mathrm{Mpc}^{-1}$).

Note that L2 is not necessarily better than L1 in matching the HF simulations. L1 matches the HF power spectrum extremely well at large scales, while L2 performs better at small scales. Therefore, the accuracy of the different simulations is not in a monotonically increasing sequence. Thus, the Kennedy & O'Hagan (2000) method we used in Ho et al. (2022) cannot be directly applied to this example.

Fig. 3 shows our experimental design in the input parameter space, corresponding to the prior range of

$$\begin{aligned}
&\Omega_0 \sim \mathcal{U}(0.24, 0.4);\\
&\Omega_b \sim \mathcal{U}(0.04, 0.06);\\
&h \sim \mathcal{U}(0.61, 0.73);\\
&A_s/10^{-9} \sim \mathcal{U}(1.7, 2.5);\\
&n_s \sim \mathcal{U}(0.92, 1),
\end{aligned} \tag{1}$$

where Ω_0 is the total matter density parameter in the Universe, Ω_b is the total baryon density parameter, h is the dimensionless Hubble parameter, A_s is the spectral amplitude, and n_s is the spectral index.

We generated 60 Latin hypercube samples using max–min sliced latin hypercube (Ba, Myers & Brenneman 2015), including 20 slices with 3 samples in each slice. We will discuss sliced latin hypercube design (SLHD) in Section 4.1. SLHD partitions the design into several equal slices (or blocks). Each slice itself is also a Latin hypercube design, as well as the whole design. We thus choose one of the Latin hypercube slices as our HF input. By using SLHD, we can avoid the design points of the HF node clustered in the corner of the prior volume. We ran L1 and L2 nodes using the same cosmological parameters (although this is not required by the GMGP from Ji et al. 2021).

We summarize the notation used in this paper in Table 2.

3 EMULATION

Emulation predicts the output from expensive cosmological simulations. First, a handful of simulations are run at carefully chosen experimental design points as a training set. Next, a surrogate model (an emulator) fits the prepared training set to predict simulation output. The trained emulator will be a proxy for the simulation results, allowing for inexpensive evaluation of a likelihood function.

In Section 3.1, we will briefly review emulation using a Gaussian process. Section 3.2 will review how we can extend the Gaussian process emulator to model simulations from different qualities using a multifidelity emulator, `MFEmulator`. Our earlier multifidelity

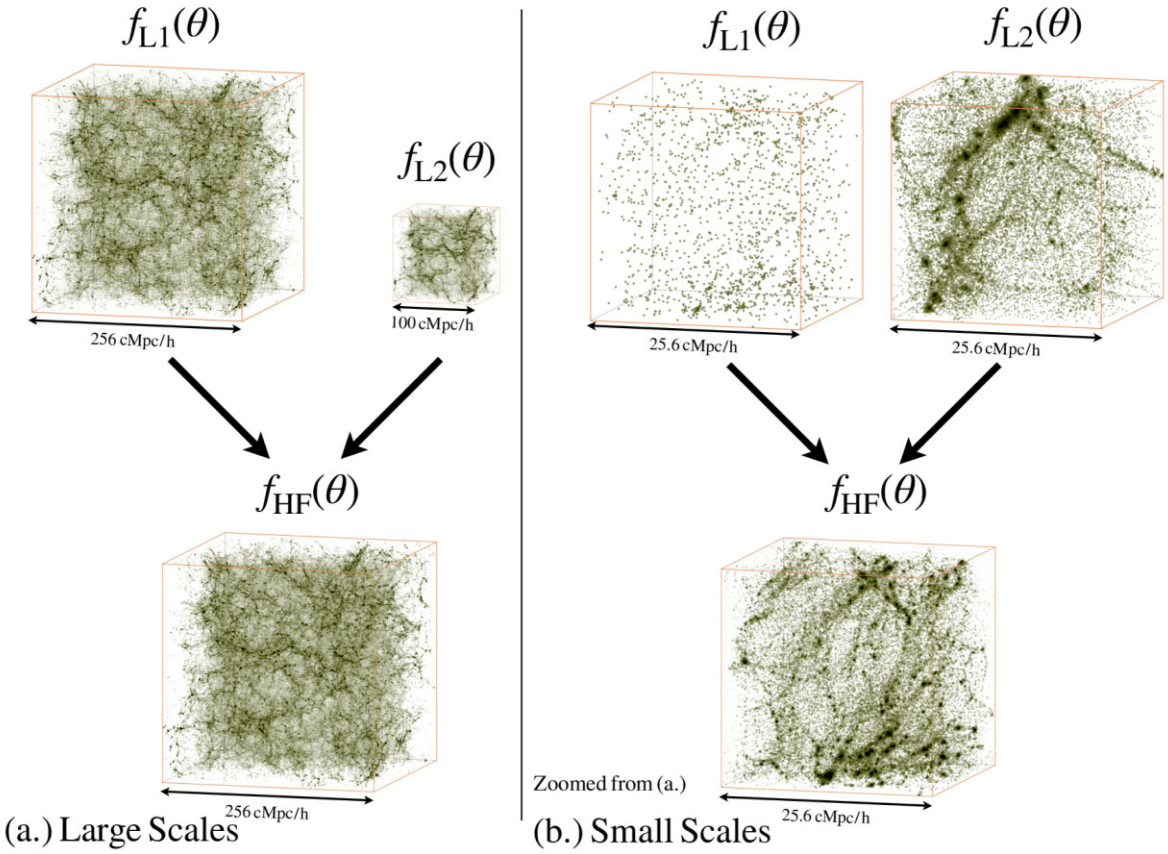

Figure 1. Illustration of the MF-Box framework and the dark-matter only simulations performed at $z = 0$. MF-Box provides a emulation framework to connect power spectra (denoted as $f(\theta)$, where θ is the input cosmology) from LF simulations (L1 and L2) to HF simulations (HF), providing an efficient emulation framework in predicting HF power spectra using only a few HF simulations augmented with many LF simulations with various volumes. ρ is a learnable multiplicative resolution correction parameter, and δ is a learnable additive resolution correction parameter. Details of the MF-Box model can be found in Section 3.2.3. The particle loads and box sizes for each simulation are listed in Table 1. (a) Large-scale structures of each simulation are shown. Simulations L1 and L2 have the same particle load ($N_p = 128$), but L1 has a smaller box size (100 Mpc h^{-1}). As a result, the large scales of L1 resemble those of the HF simulation, while L2 lacks the necessary large-scale information to match HF. (b) Zoomed-in view (25.6 Mpc h^{-1}) of the small scales from (a) L1 lacks structures due to the sparsity of particles at this scale, whereas L2 captures more structures by utilizing a smaller box size. As a result, L1 resembles HF at small scales due to its finer mass resolution.

technique based on the KO method (Kennedy & O'Hagan 2000) will be reviewed in Section 3.2.1. Section 3.2.2 will review an extension of the KO method based on a deep Gaussian process and NARGP (Perdikaris et al. 2017). Section 3.2.3 describes a graphical-model Gaussian process model (GMGP) (Ji et al. 2021), an extension of NARGP to allow more than one node in the same fidelity.

3.1 Gaussian process emulator

A Gaussian process (GP) regression model (Rasmussen & Williams 2005) is widely used as a cosmological emulator. A GP provides closed-form expressions for predictions. In addition, a GP naturally comes with uncertainty quantification, which is handy for inference framework and Bayesian optimization. In emulation, a GP can be seen as a Bayesian prior for the simulation response. It is a prior because the emulator model is chosen to ensure smoothness in the simulation response *before* data are collected (Santner et al. 2003).

Let $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$ be the input cosmologies for the simulator, and $f(\boldsymbol{\theta})$ be the corresponding output summary statistic. This work assumes that the summary statistic is the non-linear matter power spectrum. A GP regression model is a prior on the response surface of our simulated matter power spectrum:

$$p(f) = \mathcal{GP}(f; \mu, k), \tag{2}$$

where $\mu(\boldsymbol{\theta}) = \mathbb{E}[f(\boldsymbol{\theta})]$ is the mean function, and $k(\boldsymbol{\theta}, \boldsymbol{\theta}') = \text{Cov}[f(\boldsymbol{\theta}), f(\boldsymbol{\theta}')]$ is the covariance kernel function. The mean function is usually assumed to be a constant or zero mean unless there is prior knowledge about the mean function. In this work, we assume a zero mean function. The covariance kernel function is typically chosen as a squared exponential function (radial basis function, RBF) to return a smooth response surface.

Suppose we run the simulations at n carefully chosen input cosmologies, $\mathcal{D} = \{\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_n\}$, and we compress each simulation into the corresponding matter power spectrum, $\boldsymbol{y} = \{f(\boldsymbol{\theta}_1), \cdots, f(\boldsymbol{\theta}_n)\}$. Conditioning on this training data and optimizing the hyperparameters using maximum-likelihood estimation, we can get the predictive distribution of f at a new input cosmology $\boldsymbol{\theta}_*$, $f_* = f(\boldsymbol{\theta}_*)$, through a

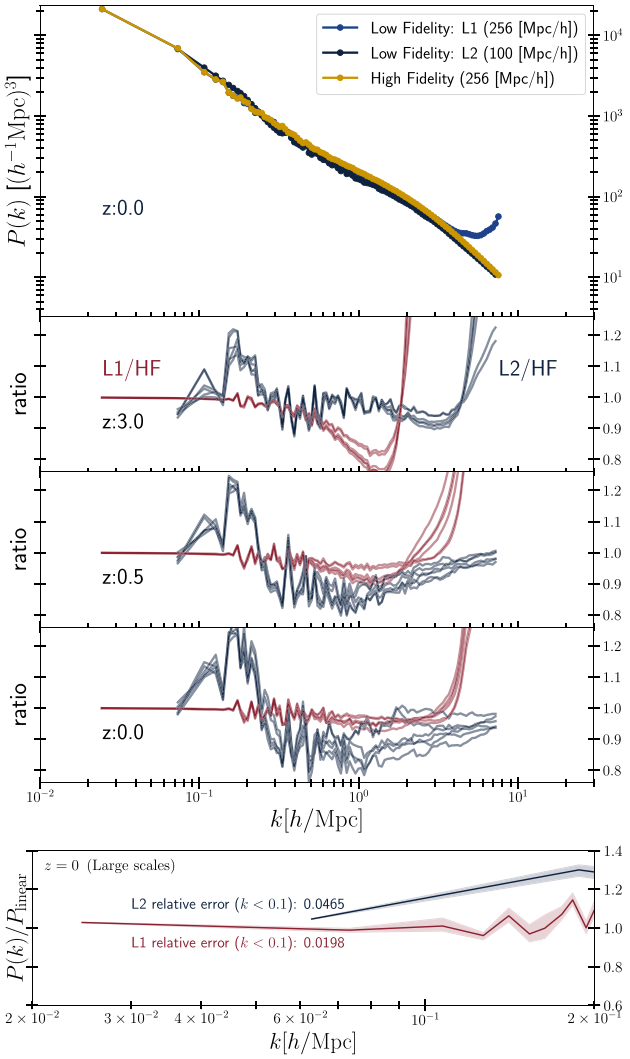

Figure 2. Matter power spectra from dark-matter only MP-Gadget simulations with various fidelities, conditioning on the same cosmology. The top panel shows the power spectra from a large-box LF (L1; blue), a small-box LF (L2; black), and a large-box HF simulations (HF; yellow). The numeric values for different fidelities of simulations are tabulated in Table 1. The second, third, and bottom panels show the ratios of L1/HF (red) and L2/HF (black) simulations, conditioned on different redshift bins, $z = 3.0,\ 0.5,\ 0$. Bottom panel: We also show the ratio between (L1, L2) and the linear theory power spectrum from CLASS at large scales. The solid lines show the median and shaded areas show the 68 per cent quantiles across 60 different cosmologies.

closed-form expression

$$p(f_* \mid \boldsymbol{y}_*, \mathcal{D}, \boldsymbol{\theta}) = \mathcal{N}(f_* \mid \mu_*(\boldsymbol{\theta}_*), \sigma_*^2(\boldsymbol{\theta}_*)), \tag{3}$$

where the mean and variance are

$$\begin{aligned}\mu_*(\boldsymbol{\theta}_*) &= \boldsymbol{k}(\boldsymbol{\theta}_*, \mathcal{D})^\intercal \mathbf{K}(\mathcal{D})^{-1}\boldsymbol{y};\\ \sigma_*^2(\boldsymbol{\theta}_*) &= k(\boldsymbol{\theta}_*, \boldsymbol{\theta}_*) - \boldsymbol{k}(\boldsymbol{\theta}_*, \mathcal{D})^\intercal \mathbf{K}(\mathcal{D})^{-1}\boldsymbol{k}(\boldsymbol{\theta}_*, \mathcal{D}).\end{aligned} \tag{4}$$

The vector $\boldsymbol{k}(\boldsymbol{\theta}_*, \mathcal{D}) = [k(\boldsymbol{\theta}_*, \boldsymbol{\theta}_1), \cdots, k(\boldsymbol{\theta}_*, \boldsymbol{\theta}_n)]$ represents the covariance between the new input cosmology, $\boldsymbol{\theta}$, and the training data. The matrix $\mathbf{K}(\mathcal{D})$ is the covariance of the training data.

Although we do not explicitly state this in the notation, we let $f(\boldsymbol{\theta})$ be a single-value output. If the target summary statistic is a vector, we let the Gaussian process model each bin separately. It will be more apparent why we make this modelling decision in later sections (Section 3.2). The primary reason is that the correlation between LF and HF summary statistics changes depending on the scales. The multifidelity method can only capture-scale dependence if we model the scales separately.[2]

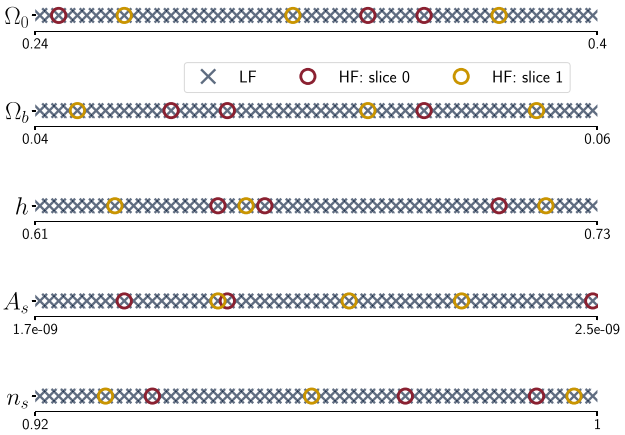

Figure 3. Experimental design of LF and HF simulations in this work. The prior volume is chosen to be the same as EuclidEmulator2 (Euclid Collaboration et al. 2021). Crosses (black) are the input parameters for the LF simulations (both L1 and L2). Circles (red and yellow) are the parameters for HF simulations, which is a subset of the LF experimental design. We use max–min Sliced Latin Hypercube (SLHD) (Ba et al. 2015) for the LF design, containing 20 slices with three samples in each slice. Red and Yellow circles show two of the slices, which we select to be the input parameters for HF simulations.

Table 2. Notations and definitions.

Notation	Description
HF	HF
LF	LF
$\boldsymbol{\theta}$	Input cosmological parameters
$f(\boldsymbol{\theta})$	Summary statistics (matter power spectrum in this work) corresponding to input parameters.
N_p	Number of particles per box side
AR1	Autoregressive GP (Kennedy & O'Hagan 2000)
NARGP	Non-linear autoregressive GP (Perdikaris et al. 2017)
GMGP	Graphical GP (Ji et al. 2021)
`MFEmulator`	Multifidelity cosmological emulator Ho et al. (2022)
`MF-Box`	Multifidelity cosmological emulator with different box sizes in LF.

3.2 Multifidelity emulation

We briefly recap the multifidelity emulation framework we proposed in Ho et al. (2022). We will first review the Kennedy–O'Hagan model (autoregressive GP; AR1) (Kennedy & O'Hagan 2000) and NARGP (non-linear autoregressive GP) (Perdikaris et al. 2017) in Sections 3.2.1 and 3.2.2, respectively. We do not change our

[2]An alternative way is to apply a co-kriging kernel to model the covariance for each vector element. We do not do that in this work because we found the single-output GP is enough for our cosmological emulation purpose, so there is no need to introduce another layer of complexity.

AR1 and NARGP modelling presented in Ho et al. (2022), except we simplified the notations to only two fidelities. Finally, we will introduce the GMGP model (Ji et al. 2021), combining simulations from different box sizes.

3.2.1 *Kennedy–O'Hagan method*

Kennedy & O'Hagan (2000) proposed a linear autoregressive GP to model the response surfaces of a sequence of computer codes with increasing fidelity. For simplicity, we assume there are only two fidelities: dark-matter only simulations with fewer particles in LF and with more particles in HF.

Let $\{\boldsymbol{y}_{\rm LF}, \boldsymbol{y}_{\rm HF}\}$ be the matter power spectrum in the training set, where $\boldsymbol{y}_{\rm LF} = \{f_{\rm LF}(\boldsymbol{\theta}_i^{\rm LF})\}_{i=1}^{n_{\rm LF}}$ and $\boldsymbol{y}_{\rm HF} = \{f_{\rm HF}(\boldsymbol{\theta}_i^{\rm HF})\}_{i=1}^{n_{\rm HF}}$. Here, $n_{\rm LF}$ and $n_{\rm HF}$ are the number of simulations in the LF and HF, respectively. The KO method models the multifidelity emulator as

$$f_{\rm HF}(\boldsymbol{\theta}) = \rho \cdot f_{\rm LF}(\boldsymbol{\theta}) + \delta(\boldsymbol{\theta}), \tag{5}$$

where ρ (the scale parameter) is a trainable parameter describing the amount of common behaviour in LF and HF response surfaces. $\delta(\boldsymbol{\theta})$ is a GP that models the remaining bias, modelling the variability that cannot be captured by correlating LF to HF. In the context of the matter power spectrum, the $\rho \cdot f_{\rm LF}(\boldsymbol{\theta})$ term dominates at the large scales describing the two-halo term while $\delta(\boldsymbol{\theta})$ dominates at the small scales describing the one-halo term.

We normalize the matter power spectra into a logarithmic scale. The sample mean is subtracted from the LF log power spectra to keep the output close to zero, while the HF log power spectra are passed directly to the training:

$$\begin{aligned} \boldsymbol{y}_{\rm LF} &\leftarrow \log \boldsymbol{y}_{\rm LF} - \mathbb{E}[\log \boldsymbol{y}_{\rm LF}]; \\ \boldsymbol{y}_{\rm HF} &\leftarrow \log \boldsymbol{y}_{\rm HF}. \end{aligned} \tag{6}$$

Not subtracting the mean spectrum of HF simulations is a compromise decision. Our benchmark multifidelity emulator uses only three HF samples, and the sample mean of three power spectra will often deviate substantially from the true mean spectrum. Instead, we entirely rely on the bias term, $\delta(\boldsymbol{\theta})$, to compensate for the deviation caused by not subtracting the mean.

As mentioned in Ho et al. (2022), the ρ parameter has to be scale-dependent (as a function of k) to model the scale-dependent correlation between high and LF. Here, we use the same method as Ho et al. (2022), where we assume equation (5) is a single-output GP model and build a KO model for each k bin of the data. In this way, we can model ρ as a function of k.

We also assign different KO models to different redshifts. We note that it is possible to assume a smooth function to model $\rho(k, z)$, and we may examine this in future work.

3.2.2 *Non-linear autoregressive Gaussian process*

Another multifidelity method we used in Ho et al. (2022) is the non-linear autoregressive GP, or NARGP, developed by Perdikaris et al. (2017). NARGP is a modification of the KO method to allow non-linearity in the scale parameter, ρ, through a deep GP (Damianou & Lawrence 2013). In cosmic emulators, it means that we allow ρ to vary as a function of cosmology.

Let $f_{\rm HF}(\boldsymbol{\theta})$ be the HF and $f_{\rm LF}(\boldsymbol{\theta})$ be the LF power spectra as functions of cosmology, $\boldsymbol{\theta}$. NARGP models the multifidelity problem as

$$f_{\rm HF}(\boldsymbol{\theta}) = \rho(\boldsymbol{\theta}, f_{\rm LF}(\boldsymbol{\theta})) + \delta(\boldsymbol{\theta}), \tag{7}$$

Here, ρ is modelled as a GP and is a function of the cosmologies, $\boldsymbol{\theta}$, and the output from the previous fidelity, $f_{\rm LF}(\boldsymbol{\theta})$. We follow the approximation made in Perdikaris et al. (2017) to simplify the computation of a deep GP to two separate GPs. The approximation is done by replacing the $f_{\rm LF}(\boldsymbol{\theta})$ with its posterior, $f_{*,\rm LF}(\boldsymbol{\theta})$. Equation (7) can thus be further reduced to a regular GP with a kernel function K:

$$f_{\rm HF} \sim \mathcal{GP}(0, K) \tag{8}$$

with

$$K(\boldsymbol{\theta}, \boldsymbol{\theta}') = K_\rho(\boldsymbol{\theta}, \boldsymbol{\theta}') \cdot K_f(f_{*,\rm LF}(\boldsymbol{\theta}), f'_{*,\rm LF}(\boldsymbol{\theta}')) + K_\delta(\boldsymbol{\theta}, \boldsymbol{\theta}'). \tag{9}$$

We integrate the bias GP and the scale parameter GP here into one single GP with a composite kernel. Each kernel, $(K_\rho, K_{\rm f}, K_\delta)$, is a squared exponential kernel. K_δ models the bias term, and the scale parameter GP is factorized into the $K_{\rm f}$, modelling the covariance between LF output posteriors. K_ρ models the cosmological dependence of ρ.

3.2.3 *Graphical multifidelity Gaussian process*

Here, we briefly explain a new multifidelity model using a GMGP, first introduced in Ji et al. (2021). A similar approach is the multi-information source method (Poloczek et al. 2017), which allows multiple LF nodes (information sources) to resolve a single HF truth. However, we find the model in Ji et al. (2021) is methodologically closer to what we applied before in Ho et al. (2022), and so use this technique for our emulation problem for LF nodes with different box sizes.

The graphical GP model (Ji et al. 2021) utilizes a directed acyclic graph to model multifidelity data. Instead of assuming the fidelities of a simulation code form a monotonically increasing sequence in accuracy, a GMGP allows the fidelities to have a directed-in tree structure. Ji et al. (2021) have a thorough mathematical description for applying GMGP in an arbitrarily directed in-tree structure. Thus, each HF node has more than one corresponding LF node, a common situation as there are many ways to approximate HF simulations.

We use the simplest case of the tree structure, illustrated in Fig. 1, with two LF nodes and one HF node. In the case of N-body simulations, one may vary not only the number of particles, but also the box size of the simulation. Thus, we can use an LF simulation with a smaller box size to improve emulation at the HF node. We will call this tree '`MF-Box`' throughout the rest of the paper. In the following text, we will assume L1 is the LF node that has 128^3 particles. L2 has the same number of particles as L1 but a smaller box size (100 Mpc h^{-1}), and HF is the HF node with 512^3 particles and the same box size as L1 (Table 1).

The deep GMGP model (dGMGP), we use from Ji et al. (2021) is an extension of NARGP, where Ji et al. (2021) implemented a specific kernel structure allowing LF information from multiple nodes to be passed to the HF node.[3] For the directed graph in Fig. 1, the dGMGP model can be written as

$$f_{\rm HF}(\boldsymbol{\theta}) = \rho(\{f_{\rm t}(\boldsymbol{\theta}) : t \in L1, L2\}, \boldsymbol{\theta}) + \delta(\boldsymbol{\theta}). \tag{10}$$

Here, we pass the cosmologies $\boldsymbol{\theta}$ and the outputs from L1 and L2 to the ρ function. We make the same approximation as in Section 3.2.2, so we can train the deep GP recursively: We first train the LF

[3] Since we found NARGP outperformed AR1 in Ho et al. (2022) for the matter power spectrum case, we will use dGMGP instead of the GMGP extended from the AR1 model.

emulators on L1 and L2, respectively. Then, we sample the output posteriors from the L1 and L2 emulators and use them as the training input for equation (10).

Similar to NARGP, we use a composite kernel for the HF GP in the dGMGP:

$$K_{\rm dGMGP}(\boldsymbol{\theta}, \boldsymbol{\theta}') = K_\rho(\boldsymbol{\theta}, \boldsymbol{\theta}') \cdot K_f(f_{*,\rm LF}(\boldsymbol{\theta}),\, f_{*,\rm LF}(\boldsymbol{\theta}')) + K_\delta(\boldsymbol{\theta}, \boldsymbol{\theta}'), \tag{11}$$

where the above expression is the same as equation (9) except that K_f takes the outputs from both L1 and L2 emulators as inputs,

$$K_f(f_{*,\rm LF}(\boldsymbol{\theta}),\, f_{*,\rm LF}(\boldsymbol{\theta}')) = K_{\rm linear}(f_{*,\rm LF}(\boldsymbol{\theta}),\, f_{*,\rm LF}(\boldsymbol{\theta}')) + K_{\rm rbf}(f_{*,\rm L1}(\boldsymbol{\theta}),\, f_{*,\rm L1}(\boldsymbol{\theta}')) \cdot K_{\rm rbf}(f_{*,\rm L2}(\boldsymbol{\theta}),\, f_{*,\rm L2}(\boldsymbol{\theta}')). \tag{12}$$

Here, $K_{\rm rbf}$ is a radial basis kernel, and $K_{\rm linear}$ is a linear kernel, which can be expressed more explicitly as

$$K_{\rm linear}(f_{*,\rm LF},\, f'_{*,\rm LF}) = \sigma_1^2 f_{*,\rm L1} f'_{*,\rm L1} + \sigma_2^2 f_{*,\rm L2} f'_{*,\rm L2},$$

where σ_1^2 and σ_2^2 are the hyperparameters of the linear kernel. A linear kernel in a Gaussian process is equivalent to a Bayesian linear regression.[4] The multiplication in the kernel operation means an 'AND' operation, showing high covariance only if both kernels have high values. The addition operator means an 'OR' operation, indicating the final covariance is high if either of the kernels gives a high value. The intuition here is that the linear kernel encodes the linear regression part while the multiplication of RBF kernels encodes the non-linear transformation from L1 and L2 nodes to the HF node

4 SAMPLING STRATEGY FOR HF SIMULATIONS

This section describes the method used for selecting the input parameters for our HF training simulations. Following Ji et al. (2021), we employ an SLHD (Qian 2012; Ba et al. 2015) to assign input parameters for the HF nodes. Each slice (or subset) in an SLHD is a Latin hypercube and thus can be served as the design points for the HF node. This approach offers a less computationally intensive and more straightforward implementation compared to the grid search method utilized in our previous work (Ho et al. 2022). The details of SLHD will be discussed in Section 4.1, and our process for selecting the optimal HF design from the SLHD will be discussed in Section 4.2.

4.1 Sliced Latin hypercube design

SLHD is a type of Latin hypercube that can be partitioned by slices or blocks, each of which contains an equal number of design points. Each slice is itself a Latin hypercube. SLHD ensures the space-filling property both in the whole design and in each slice. Therefore, SLHD is an intuitive choice for a multifidelity problem.

Suppose we have an SLHD for the LF node. We can use one of the slices to generate simulations for the HF node, which ensures that both the LF and HF nodes are in Latin hypercubes. Another advantage of SLHD is that we can directly obtain a nested experimental design where the LF samples form a superset of the HF samples, i.e. $\boldsymbol{\theta}_{\rm HF} \subset \boldsymbol{\theta}_{\rm LF}$. As mentioned in Kennedy & O'Hagan (2000), a nested design is an efficient training set for a multifidelity model because it allows us to obtain an accurate posterior $f_{\rm LF}(\theta)$ at location θ without interpolating at the LF.

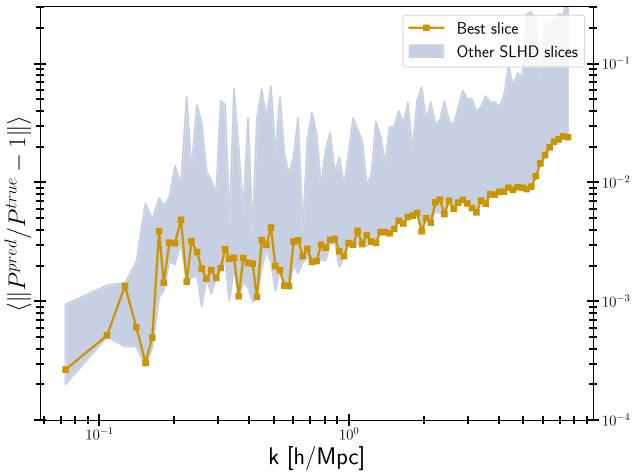

Figure 4. `MF-Box`'s emulation errors, averaged over redshift bins and test simulations, using 60 L1, 60 L2, and 3 HF (see Table 1). Here, we show the emulation minimum and maximum errors using different slices from SLHD (blue-shaded area), and the best slice found by the grid search method is labelled as yellow.

SLHD, initially proposed by Qian (2012), is a technique developed for applying the Latin hypercube design to categorical variables. Ba et al. (2015) later developed an efficient method for constructing optimal SLHD designs. The number of categories for categorical variables is usually fixed based on qualitative properties, making it challenging to apply a Latin hypercube design to such variables. However, SLHD addresses this challenge and enables the use of Latin hypercube designs with categorical variables. In SLHD, a Latin hypercube is divided into equal slices along the dimensions associated with categorical variables, while non-categorical dimensions are still sampled with ordinary Latin hypercube sampling. The usage of SLHD in the context of modelling the multifidelity problem was demonstrated in Ji et al. (2021). Furthermore, SLHD has also been employed in cosmology, specifically by the Dark Emulator (Nishimichi et al. 2019).

For implementation, we use the maximin SLHD package, `maxminSLHD`,[5] in `R` (Ba et al. 2015). We set the number of design points to three for each slice and the number of slices to 20. In total, we have 60 design points. We assign the SLHD with 60 points to LF and select one slice as our HF design. We use 60 LF points in this work because we learned in Ho et al. (2022) that ∼50 simulations are enough for a five dimensional emulation problem.

4.2 Selecting the optimal slice

Slices in SLHD are Latin hypercubes in smaller sizes. In principle, any slice should produce reasonably good emulation, as the points in each slices span parameter space.

However, in practice, some slices still perform somewhat better than others, as shown in Fig. 4. We use a procedure similar to our grid search approach in Ho et al. (2022) to avoid choosing the worst slice. The procedure is described as follows:

(i) prepare SLHD for LF simulation suite;

[4]See the kernel cookbook: https://www.cs.toronto.edu/~duvenaud/cookbook/.

[5]https://rdrr.io/cran/SLHD/man/maximinSLHD.html

(ii) build LF only emulators (LFEmu) for each slice, compute the interpolation error for each LFEmu, testing solely on the LF simulation suite; and

(iii) select the slice which can best minimize the interpolation error.

Note that we do not use any HF simulations in the above procedure. The selection entirely relies on the LF simulation suite. The underlying assumption is that the interpolation error of the LF node is correlated with the interpolation error of the HF node. We labelled the selected slice in Fig. 3. We will use the best slice as our HF training set for the results in Section 6.

To summarize, SLHD is a special kind of LHD, with each slice in the SLHD being a Latin hypercube as well as the whole design. We thus can assign HF nodes with a slice (or slices) of SLHD, making both LF and HF nodes Latin hypercubes. In the end, we describe a procedure to avoid choosing the worst slice for training an `MFEmulator`.

5 COMPUTATIONAL BUDGET ESTIMATION

In this section, we present our approach to quantifying the optimal allocation of simulation budgets across different fidelities. Building upon the error bounds established in Ji et al. (2021), we have made modifications to adapt them to our specific context, as described in Section 5.1. We approximate the emulation errors of our `MF-Box` using the form of Ji et al. (2021) and empirically infer the error function of the emulator for various training designs, denoted as ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$). Our objective is to utilize this empirical error function to determine the most cost-effective strategy for assigning LF and HF simulations in order to achieve optimal accuracy.

In Section 5.1, we present an approximate error function for our `MF-Box` emulator in predicting HF simulation outputs. Next, in Section 5.2, we show the analysis for assigning optimal computational budgets to LF and HF simulations, under the assumption that the emulator error follows the approximate error function. In Section 5.3, we empirically estimate the approximate error function of the `MF-Box` by analysing the average emulator errors obtained from 144 distinct `MF-Box` training results. Finally, we determine the optimal number of LF and HF simulations required for achieving accurate power spectra emulation using the `MF-Box` approach.

5.1 Error bounds for Gaussian process emulators

Ji et al. (2021) presents an error bound for a multifidelity emulator, and for the case of two LF nodes, the form is given by $\sim \mathcal{O}(\rho_{\rm L1} \cdot n_{\rm L1}^{-\frac{\nu_{\rm L1}}{d}} + \rho_{\rm L2} \cdot n_{\rm L2}^{-\frac{\nu_{\rm L2}}{d}} + n_{\rm HF}^{-\frac{\nu_{\rm HF}}{d}})$, where ($\rho_{\rm L1}$, $\rho_{\rm L2}$) are the scale parameters for the L1 and L2 nodes, respectively. ($\nu_{\rm L1}$, $\nu_{\rm L2}$, $\nu_{\rm HF}$) are positive spectral indices, and ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$) represent the number of training simulations at the L1, L2, and HF nodes, respectively. While this bound does not directly apply to our case, we utilize the form of the bound as an approximate model for the `MF-Box` error and empirically determine the parameters by fitting them to the `MF-Box` emulation results using different multifidelity designs, i.e. varying combinations of ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$).

The equation below represents the error function of the `MF-Box` emulator we want to infer. Note that our discussion primarily focuses on the emulation error when predicting 'HF' power spectra. This emphasis aligns with the core objective of `MF-Box`, which is to correct the resolution of LF simulations for accurate predictions of their HF counterparts.

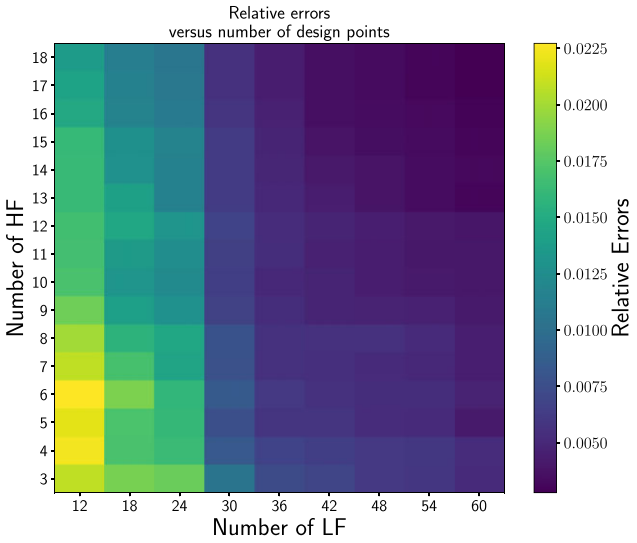

Figure 5. Relative errors plotted against the number of LF and HF design points in an `MF-Box` emulator. Here, LF refers to the combined number of L1 and L2 points, where LF $=n_{\rm L1} = n_{\rm L2}$. The plot reveals a trend of decreasing errors as the number of LF training simulations increases. However, due to the limited number of HF points compared to LF points, the decreasing trend is relatively modest.

$$
\begin{aligned}
&\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left|\frac{f_{\rm HF}(\boldsymbol{\theta}_i) - m_{f_{\rm HF}}(\boldsymbol{\theta}_i)}{f_{\rm HF}(\boldsymbol{\theta}_i)}\right| \\
&\approx \tilde{\Phi}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}) \\
&= \eta \cdot \left(\rho_{\rm L1} \cdot n_{\rm L1}^{-\frac{\nu_{\rm L1}}{d}} + \rho_{\rm L2} \cdot n_{\rm L2}^{-\frac{\nu_{\rm L2}}{d}} + n_{\rm HF}^{-\frac{\nu_{\rm HF}}{d}}\right), \qquad (13)
\end{aligned}
$$

where $N = 10$ test simulations in a Latin hypercube are used to average the emulation relative error. The emulator error function $\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$ represents the average relative error of the `MF-Box` as a function of the number of simulations in L1, L2, and HF nodes. To estimate this error function, we have already averaged the emulation error across k bins, enabling us to obtain an approximation of the error as a function of the design points ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$). Then, we infer the parameters of this error function from the `MF-Box` emulation results, as denoted by the $\approx$ sign in equation (13). The normalization factor of the functional form in equation (13) is determined by the free parameter η.

An important term in equation (13) is the one describing how the error scales with an increasing number of simulations, $n_t^{-\frac{1}{d}}$, where $t \in$ L1, L2, HF. This scaling term comes from the fact that the fill distance is proportional to $\mathcal{O}(n_t^{-\frac{1}{d}})$, where d is the number of dimensions in a space-filling design (Wendland 2004).

To determine the parameters of $\tilde{\Phi}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$, we employ MCMC inference based on 144 distinct `MF-Box` emulators that were trained with varying numbers of ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$). Specifically, we generated `MF-Box` emulators using [12, 18, 24,..., 60] L1/L2 points and [2, 3,...,18] HF points, resulting in a total of 144 emulators. For simplicity, we only considered cases where the number of simulations in L1 and L2 nodes was equal, i.e. $n_{\rm L1} = n_{\rm L2}$, as the costs of L1 and L2 nodes are similar, therefore, choosing between them is not important. To simplify the notation, we employ $n_{\rm LF}$ to represent the number of training points in both the L1 and L2 nodes. Fig. 5 presents the

Table 3. MCMC analysis of equation 13: $\frac{1}{N}\sum_{i=1}^{N}\left|\frac{f_{\rm HF}(\boldsymbol{\theta}_i)-m_{f_{\rm HF}}(\boldsymbol{\theta}_i)}{f_{\rm HF}(\boldsymbol{\theta}_i)}\right| = \Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}) \approx \eta\cdot\left(\rho_{\rm L1}\cdot n_{\rm L1}^{-\frac{\nu_{\rm L1}}{d}} + \rho_{\rm L2}\cdot n_{\rm L2}^{-\frac{\nu_{\rm L2}}{d}} + n_{\rm HF}^{-\frac{\nu_{\rm HF}}{d}}\right)$. The notation $\{\Phi(n_{{\rm L1},j}, n_{{\rm L2},j}, n_{{\rm HF},j})\}_{j=1}^{144}$

Parameters	Prior	Posterior (50 per cent)	Posterior (25 per cent, 75 per cent)
η	Normal($\mu = \text{Mean}(\{\Phi_j\}_{j=1}^{144}), \sigma^2 = \text{Var}(\{\Phi_j\}_{j=1}^{144})$)	0.0308	(0.0290, 0.0327)
$\nu_{\rm HF}$	LogNormal ($\mu = 0, \sigma = 1$)	9.80	(9.44, 10.2)
$\nu_{\rm L1}$	LogNormal ($\mu = 0, \sigma = 1$)	5.49	(5.33, 5.67)
$\nu_{\rm L2}$	LogNormal ($\mu = 0, \sigma = 1$)	5.49	(5.33, 5.67)
$\rho_{\rm L1}$	Normal ($\mu = 1, \sigma = 1$)	4.53	(3.97, 5.08)
$\rho_{\rm L2}$	Normal ($\mu = 1, \sigma = 1$)	4.54	(3.97, 5.10)

Notes. The notation$\{\Phi(n_{{\rm L1},j}, n_{{\rm L2},j}, n_{{\rm HF},j})\}_{j=1}^{144}$means all 144 `MF-Box` emulator errors used for parameter estimation. The column 'Posterior (50 per cent)' reports the medians of the posteriors of the parameters, and 'Posterior (25 per cent, 75 per cent)' reports the 25 and 75 per cent quantities of the posterior distributions.

average relative errors, $\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$, for all 144 designs under consideration.

For each pixel in Fig. 5, we compute the average emulator relative error across 10 test simulations and multiple k bins across a redshift range, $z \in [0, 0.2, 0.5, 1, 2, 3]$. To solve the parameter estimation problem, we employ MCMC inference with a Gaussian likelihood,[6]

$$\begin{aligned}&\tilde{\Phi}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})\\ &= \eta\cdot\left(\rho_{\rm L1}\cdot n_{\rm L1}^{-\frac{\nu_{\rm L1}}{d}} + \rho_{\rm L2}\cdot n_{\rm L2}^{-\frac{\nu_{\rm L2}}{d}} + n_{\rm HF}^{-\frac{\nu_{\rm HF}}{d}}\right)\\ &\sim \mathcal{N}(\mu = \Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}), \sigma^2 = \Phi_{\rm var}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})).\end{aligned} \tag{14}$$

Here, $\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$ represents the average relative errors, while $\Phi_{\rm var}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$ denotes the variance of the relative errors across 10 test simulations.

The results of our MCMC analysis, including the priors and posteriors, are summarized in Table 3. The posteriors show that $\nu_{\rm L1} \simeq \nu_{\rm L2}$ and $\rho_{\rm L1} \simeq \rho_{\rm L2}$, indicating that both L1 and L2 nodes contribute to improving the accuracy of the emulator in a similar manner. In contrast, the power-law index $\nu_{\rm HF}$ for the HF node is approximately twice as large as $\nu_{\rm L1}$ and $\nu_{\rm L2}$, suggesting that the HF node has a more pronounced impact on enhancing the emulator's accuracy compared to the LF nodes. Table 3 shows that the parameters in equation (13) are reasonably well defined. Thus, we will use the median of the posterior as point estimates for the error function for the remainder of this paper.

5.2 Optimal number of simulations per node

Equation (13) models the emulation error, $\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$, which behaves as a combination of power-law functions of the number of simulations in each node, namely LF or HF. The primary goal of an emulator is to better represent the original simulator by minimizing the prediction error, subject to a limited computational budget, denoted by C. By using $\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$, we can determine the optimal number of simulations per node, given the computational budget available for running each node.

Consider a two-fidelity emulator consisting of two LF nodes, L1 and L2, where $\rho_{\rm L1,\,L2}$ are the scale parameters and ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$) represent the number of simulations in L1, L2, and HF nodes, respectively. Our goal is to minimize the emulation error while subject to a limited budget.

$$n_{\rm L1}\cdot C_{\rm L1} + n_{\rm L2}\cdot C_{\rm L2} + n_{\rm HF}\cdot C_{\rm HF} \le C, \tag{15}$$

where we know the ratios between the costs of HF and LF nodes (L1 and L2) are $\frac{C_{\rm HF}}{C_{\rm L1}} \simeq 140$ and $\frac{C_{\rm HF}}{C_{\rm L2}} \simeq 140/1.7$, from Table 1.

The Lagrangian for optimizing the error subjecting to the cost is

$$\begin{aligned}\mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda) = \eta&\left(\rho_{\rm L1}\cdot n_{\rm L1}^{-\frac{\nu_{\rm L1}}{d}} + \rho_{\rm L2}\cdot n_{\rm L2}^{-\frac{\nu_{\rm L2}}{d}} + n_{\rm HF}^{-\frac{\nu_{\rm HF}}{d}}\right)\\ &+ \lambda(n_{\rm L1}\cdot C_{\rm L1} + n_{\rm L2}\cdot C_{\rm L2} + n_{\rm HF}\cdot C_{\rm HF} - C),\end{aligned} \tag{16}$$

Here, λ is the Lagrange multiplier. To find the optimal number of ($n_{\rm L1}$, $n_{\rm L2}$, $n_{\rm HF}$) minimizing the emulation error, we use the first-order derivative conditions of the Lagrangian,

$$\begin{aligned}\frac{\partial\mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda)}{\partial n_{\rm L1}} &= 0;\\ \frac{\partial\mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda)}{\partial n_{\rm L2}} &= 0;\\ \frac{\partial\mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda)}{\partial n_{\rm HF}} &= 0,\end{aligned} \tag{17}$$

resulting in

$$\begin{aligned}\eta\frac{\nu_{\rm L1}}{d}\rho_{\rm L1}\cdot n_{\rm L1}^{-\frac{\nu_{\rm L1}+d}{d}} &= \lambda C_{\rm L1} \Rightarrow n_{\rm L1} \propto \left(\frac{\nu_{\rm L1}\rho_{\rm L1}}{C_{\rm L1}}\right)^{\frac{d}{\nu_{\rm L1}+d}}\\ \eta\frac{\nu_{\rm L2}}{d}\rho_{\rm L2}\cdot n_{\rm L2}^{-\frac{\nu_{\rm L2}+d}{d}} &= \lambda C_{\rm L2} \Rightarrow n_{\rm L2} \propto \left(\frac{\nu_{\rm L2}\rho_{\rm L2}}{C_{\rm L2}}\right)^{\frac{d}{\nu_{\rm L2}+d}}\\ \eta\frac{\nu_{\rm HF}}{d}n_{\rm HF}^{-\frac{\nu_{\rm HF}+d}{d}} &= \lambda C_{\rm HF} \Rightarrow n_{\rm HF} \propto \left(\frac{\nu_{\rm HF}}{C_{\rm HF}}\right)^{\frac{d}{\nu_{\rm HF}+d}}.\end{aligned} \tag{18}$$

Here, the intuition is relatively straightforward: the number of simulations required is inversely proportional to the cost of each simulation at a given fidelity. However, if we observe a strong correlation between fidelities (i.e. if $\rho_{\rm L1,\,L2}$ is large), then we should use more LF simulations because they are less expensive.

To ensure that equation (18) identifies local minima instead of maxima, we can verify the positivity of the second-order derivatives

[6]We use the `PyMC` package version 4 (Salvatier, Wiecki & Fonnesbeck 2016) for the MCMC inference.

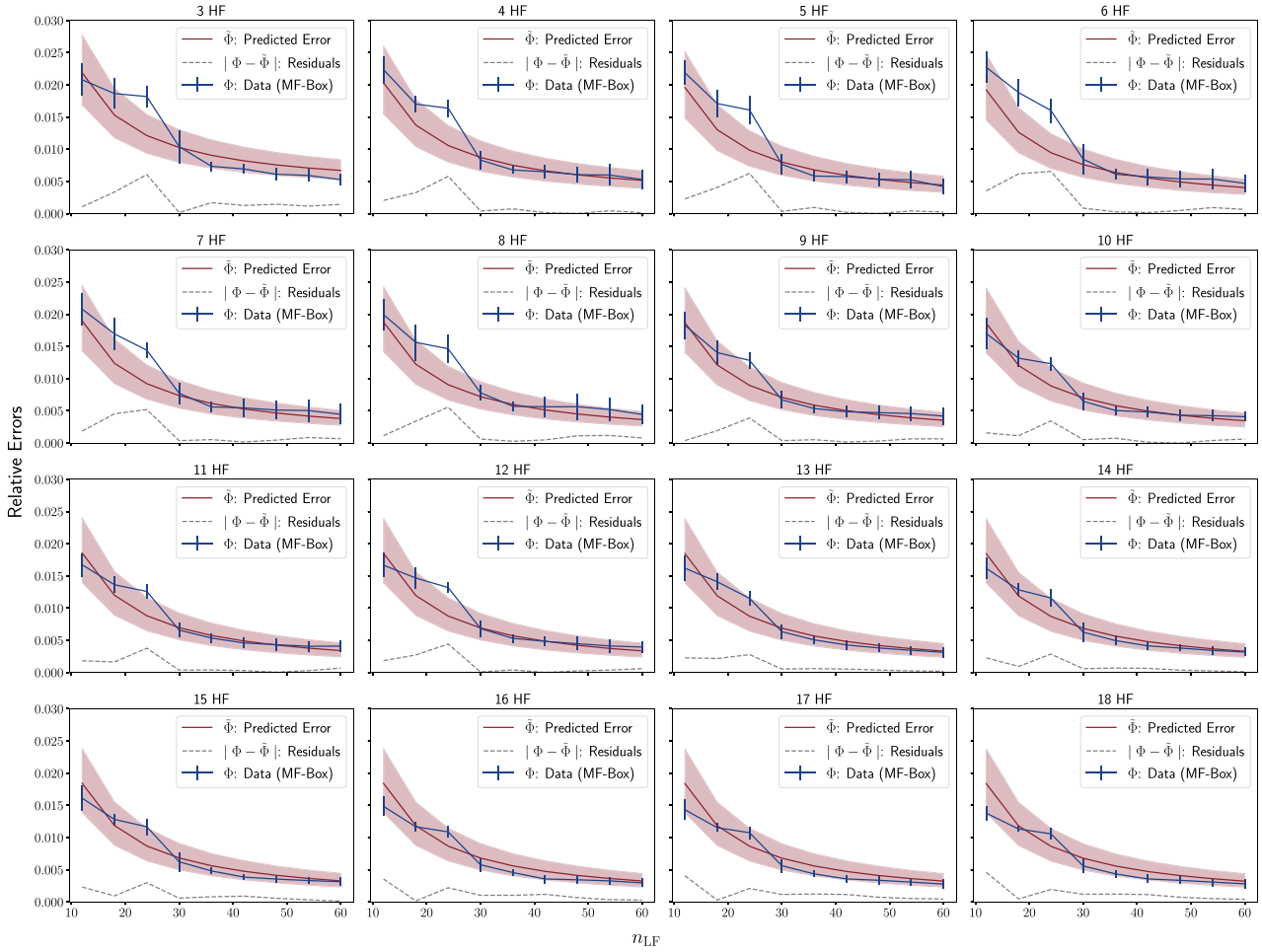

Figure 6. Inferred relative errors for all available `MF-Box` emulators are displayed. Each subplot corresponds to a fixed number of HF points (as indicated in the title) with varying LF points (on the *x*-axis). The red curves represent the median predictions (50 per cent posterior). Blue lines indicate the average relative errors obtained from the `MF-Box` emulators, while the error bars represent the standard deviation of relative errors across 10 simulations in the test set. The shaded area depicts the 25 and 75 per cent confidence interval of the predictions based on the inference results. Overall, the relative errors demonstrate a decreasing trend as the number of LF and HF points increases.

of the Lagrangian.

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda)}{\partial n_{\rm L1}^2} &= \eta\rho_{\rm L1}\frac{\nu_{\rm L1}(\nu_{\rm L1}+d)}{d^2} n_{\rm L1}^{-\frac{\nu_{\rm L1}+2d}{d}} > 0;\\
\frac{\partial^2 \mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda)}{\partial n_{\rm L2}^2} &= \eta\rho_{\rm L2}\frac{\nu_{\rm L2}(\nu_{\rm L2}+d)}{d^2} n_{\rm L2}^{-\frac{\nu_{\rm L2}+2d}{d}} > 0;\\
\frac{\partial^2 \mathcal{L}(n_{\rm L1}, n_{\rm L2}, n_{\rm HF}, \lambda)}{\partial n_{\rm HF}^2} &= \eta\frac{\nu_{\rm HF}(\nu_{\rm HF}+d)}{d^2} n_{\rm HF}^{-\frac{\nu_{\rm HF}+2d}{d}} > 0.
\end{aligned} \tag{19}$$

The parameters $(\nu_{\rm L1}, \nu_{\rm L2}, \nu_{\rm HF})$, $(\rho_{\rm L1}, \rho_{\rm L2}, \rho_{\rm HF})$, and η are all positive, while the dimension of the input space, d, must be a positive integer. Similarly, the number of simulations $(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$ must be positive integers as well. Therefore, all second-order derivatives are positive, indicating that equation (18) minimizes the emulation error.

In the special case, where $\nu \equiv \nu_{\rm LF} = \nu_{\rm HF}$, equation (18) simplifies to the optimal budget identified in Ji et al. (2021):

$$\frac{n_{\rm LF}}{n_{\rm HF}} = \left(\frac{\rho_{\rm LF} C_{\rm HF}}{C_{\rm LF}}\right)^{\frac{d}{\nu+d}}, \tag{20}$$

where the ratio of LF/HF training sample sizes is inversely proportional to the cost of each simulation per run and directly proportional to the correlation with the HF node.

5.3 Empirical estimate of the error function

In this section, we present the predicted errors of `MF-Box` obtained from our MCMC analysis. We explore the impact of different `MF-Box` designs on error predictions. Finally, we discuss the choices of the optimal number of simulations for `MF-Box` based on the analysis presented in Section 5.2.

We illustrate the predicted emulation errors in Fig. 6, categorized by `MF-Box` models with varying LF and HF points. The predictions align with the overall trend of the data, except when $n_{\rm LF}$ is low, where the limited availability of LF training points leads to suboptimal training performance.

Figs 7 and 8 depict the predicted relative errors as a function of LF and HF points, respectively. Both figures exhibit a power-law trend characterized by a negative spectral index, indicating that the error decreases as the number of training points increases. For example, in Fig. 7, the *X* LF-3 HF emulator emulators ($X \in \{12, 18, 24, 30, 36, 42, 48, 54, 60\}$) follow this trend concerning the number of

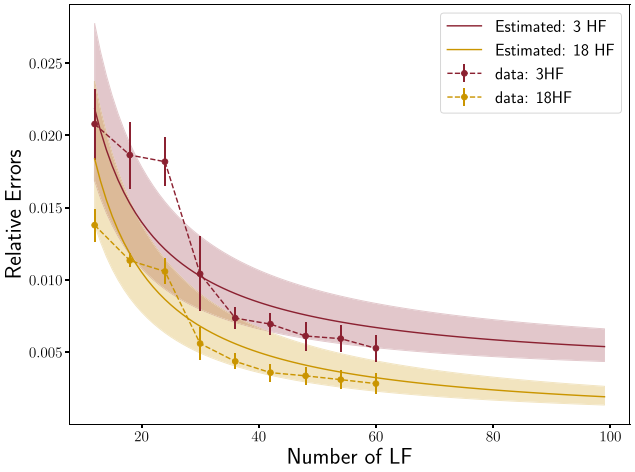

Figure 7. Inferred relative errors as a function of LF points. Shaded area shows the 25 and 75 per cent confidence interval of the prediction from the inference result.

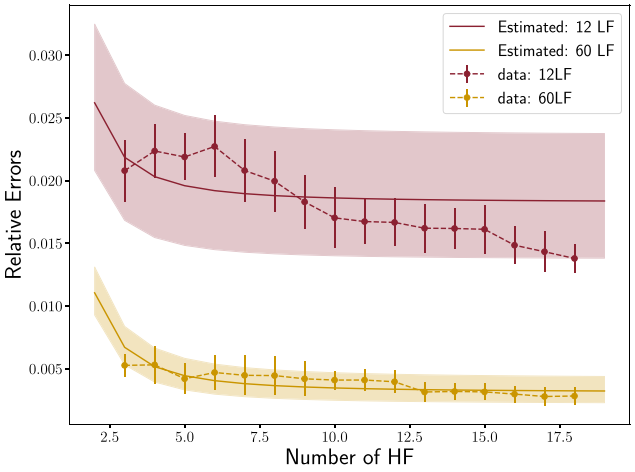

Figure 8. Inferred relative errors as a function of HF points. Shaded area shows the 25 and 75 per cent confidence interval of the prediction from the inference result.

LF points, suggesting that achieving further accuracy improvements becomes challenging once a sufficient number of LF points are used. How much the error can be reduced by increasing the number of LF points is also influenced by the correlation between LF and HF simulations, which is controlled by the ρ parameter. A higher value of ρ indicates that LF points can more effectively reduce the error.

On the other hand, incorporating additional HF points can also enhance accuracy. In Fig. 7, increasing the number of points in the HF node from 3 to 18 shifts the power-law function towards lower values, which itself follows the trend in Fig. 8. Similarly, as more HF points are included in the training, achieving further emulation accuracy becomes more challenging.

Fig. 9 displays the predicted error functions $\Phi(n_{\rm L1}, n_{\rm L2}, n_{\rm HF})$ for different `MF-Box` emulator designs. We compile these predictions to create a plot of emulator error versus budget size. The bottom left region of the plot represents the most economical budget setup, where the error is minimized relative to the allocated budget.

Based on the predictions in Fig. 9, we can determine the optimal number of simulations ($n_{\rm L1}$, $n_{\rm L2}$, and $n_{\rm HF}$) for achieving a desired level

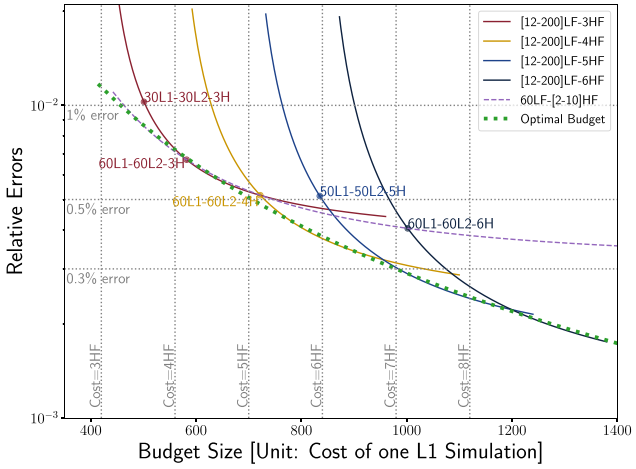

Figure 9. The predicted emulator errors as a function of the budget size, in the unit of the number of LF simulations. The predictions are based on the medians of the parameter posteriors presented in Table 3. The plot shows the predicted error functions using different combinations of LF and HF nodes. The red, yellow, blue, and black curves represent the predicted error functions with varying LF nodes and a fixed HF node ($n_{\rm HF} = 3, 4, 5, 6$). In contrast, the purple-dashed curve represents the predicted error function with varying HF nodes and a fixed LF node ($n_{\rm LF} = 60$). The green-dotted line illustrates the error function corresponding to the optimal budget (equation 21). The vertical grey-dotted lines indicate the budget size in terms of the number of HF simulations. The horizontal grey-dotted lines denote the predicted errors at the levels of (1 per cent, 0.5 per cent, 0.3 per cent).

of average accuracy. For instance, if we aim for at least 1 per cent average error, the optimal choice is ($n_{\rm L1} = 30$, $n_{\rm L2} = 30$, and $n_{\rm HF} = 3$), which corresponds to a cost of approximately 500 L1 simulations. Note that a minimum of three HF simulations (∼420 L1 simulations) is required to train an `MF-Box` in our power spectrum emulation problem. Similarly, if we aim for at least 0.5 per cent average error, the optimal setup becomes ($n_{\rm L1} = 60$, $n_{\rm L2} = 60$, and $n_{\rm HF} = 4$). However, a slightly higher cost is required for the setup with ($n_{\rm L1} = 50$, $n_{\rm L2} = 50$, and $n_{\rm HF} = 5$), which yields a similar error.

In Fig. 9, the purple-dashed curve represents the predicted error of 60 LF-[2-10] HF emulators, illustrating the trend of increasing the number of HF points while keeping a fixed number of 60 LF nodes. At the point of (60 LF, 3 HF), the error decrease exhibits a similar gradient to [12-200] LF-3 HF emulator, but it shows a steeper gradient after four HF points. This result suggests that adding more LF or HF nodes does not necessarily lead to superior performance compared to each other.

Under the assumptions outlined in Section 5.2, we can determine an optimal number of simulations ($n_{\rm L1}$, $n_{\rm L2}$,and $n_{\rm HF}$) for an `MF-Box` to achieve the best emulation accuracy within a given computational budget. The optimal ratio between the number of HF and LF simulations can be expressed as

$$n_{\rm LF}^{-\frac{\nu_{\rm LF}+d}{d}} = n_{\rm HF}^{-\frac{\nu_{\rm HF}+d}{d}} \frac{C_{\rm LF}}{C_{\rm HF}} \frac{\nu_{\rm HF}}{\rho_{\rm LF}\nu_{\rm LF}}. \quad (21)$$

Here, LF is either L1 or L2. $C_{\rm LF}$ and $C_{\rm HF}$ represent the computational cost of one simulation in the LF and HF, respectively.

In Fig. 9, the green-dotted line represents the optimal budget according to equation (21). When $n_{\rm HF} = 2.5$, the optimal number of LF simulations is ($n_{\rm L1}$, $n_{\rm L2}$) = (80, 60), which is close to our initial setup of `MF-Box` with ($n_{\rm L1} = 60$, $n_{\rm L2} = 60$, and $n_{\rm HF} = 3$). Moreover, the design of ($n_{\rm L1} = 60$, $n_{\rm L2} = 60$, and $n_{\rm HF} = 4$) is also

nearly optimal (close to the green-dotted line), as demonstrated in Fig. 9.

In summary, this section introduces an approach to model the average emulation error of `MF-Box` as a function of LF and HF points using an approximate error model based on power-law functions. Through empirical analysis of 144 `MF-Box` designs with various configurations, we have inferred this error model. We demonstrate that this empirical model can guide the selection of an optimal design within a given computational budget, facilitating the construction of accurate emulators in a resource-efficient manner.

6 RESULTS

This section will demonstrate the emulation accuracy achieved by incorporating simulations with different box sizes through `MF-Box` for correcting the resolution of LF emulators to predict HF counterparts. The emulation error in this section is computed using a hold-out test set comprising 10 HF simulations, carefully selected from a separate Latin hypercube that was not part of the training set. Here, we will use `MF-Box` to denote the emulators using the GMGP model (Ji et al. 2021) with the graph structure in Fig. 1. Section 6.1 will show how `MF-Box`'s accuracy improves by adding an L2 node in 100 Mpc h^{-1}. Section 6.2 will show how `MF-Box`'s accuracy changed as a function of L2 box size, from 100 to 256 Mpc h^{-1}. Finally, Section 6.3 shows the runtime comparison between single-fidelity emulators, `MFEmulator` (including AR1, NARGP) and `MF-Box`.

6.1 `MF-Box` accuracy (256 + 100 Mpc h^{-1})

This section shows how the emulation error changed when a suite of small-box simulations is included as a second LF node, L2, through `MF-Box`. More precisely, we use two LF nodes:

(i) L1: 128^3 simulations with 256 Mpc h^{-1};
(ii) L2: 128^3 simulations with 100 Mpc h^{-1}.

The information about the training simulations is summarized in Table 1.

Fig.10 shows the emulation error averaged over redshift bins, $z \in [0, 3]$, by using different multifidelity models, AR1, NARGP, and `MF-Box`. All three models perform similarly at large scales ($k < 2\,h\mathrm{Mpc}^{-1}$). The main difference is `MF-Box` performs better at $k \geq 2\,h\mathrm{Mpc}^{-1}$ while AR1 and NARGP have an error bump at 10 per cent level.

In the right panel of Fig. 11, we can easily see the 10 per cent error bump exists at $z = 1-3$ at small scales ($k \geq 1\,h\mathrm{Mpc}^{-1}$). The small-scale improvement in the right panel is not a surprise. The additional LF node in a smaller box (L2) brings more accurate small-scale statistics than L1, making `MF-Box` outperform AR1 and NARGP. `MF-Box` stays $\simeq 1$ per cent error within the redshift range $z \in [0, 3]$, in contrast to AR1 and NARGP where the error increases from $\simeq 1$ per cent to $\simeq 20$ per cent (from $z = 0$ to 3).

The bump in interpolation error in AR1 and NARGP at $z > 1$ is due to the feature at the initial inter-particle spacing at these redshifts, corresponding to the initial particle grid, as mentioned in Ho et al. (2022). The mean particle spacing of the initial condition appears as a delta function in the matter power spectrum at high redshift. This feature eventually disappears, erased by gravitational interactions. The L2 and HF box, however, both have a smaller mean interparticle spacing and thus show the delta function on smaller scales, beyond those we wish to emulate. Using the information the L2 simulations provide, `MF-Box` is able to maintain similar accuracy across $z \in [0, 3]$.

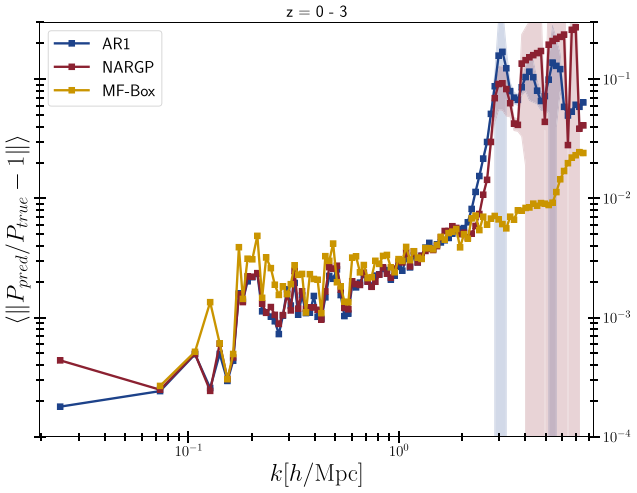

Figure 10. Relative errors averaged over $z = [0, 0.2, 0.5, 1, 2, 3]$ for different multifidelity models, AR1 (blue), NARGP (red), and `MF-Box` (yellow). The `MF-Box` model uses 60 L1 (256 Mpc h^{-1}), 60 L2 (100 Mpc h^{-1}), and 3 H (256 Mpc h^{-1}) simulations for training. Both AR1 and NARGP use 60 L1 and 3 HF for training. The shaded area is the variance among different test simulations.

The left panel of Fig. 11 shows the redshift trend at large scales, indicating no significant difference between AR1, NARGP, and `MF-Box`. The slightly worse accuracy in `MF-Box` is probably because `MF-Box` has more hyperparameters to fit, making it slightly more difficult to reach ~ 0.1 per cent accuracy.

Fig.12 shows the AR1, NARGP, and `MF-Box` accuracies as a function of the number of HF points, splitting into two redshift bins. The left panel shows the accuracy averaged over the low-redshift bins, $z \in [0, 0.2, 0.5]$, where NARGP and `MF-Box` perform similarly and outperform the AR1 model. It is not a surprise that NARGP and `MF-Box` perform similarly since `MF-Box` is an extension of NARGP.

The left panel of Fig. 12 shows that the error is almost flat as a function of HF points. In Section 5, we showed that the emulator error is a power-law function of the number of training points. Here, the emulation accuracy is likely limited by the intrinsic accuracy of our 512^3 HF simulations, so it is hard to get improvement at the sub-per cent level.[7] The right panel of Fig. 12 shows that `MF-Box` performs better than the other two models by a factor of $\sim 5-10$.

Fig.13 shows the averaged emulation error as a function of LF points. We see a mild improvement at low-redshift bins (left panel) by adding more LF points for all three models. At the higher redshift bins (right panel), AR1 and NARGP cannot be easily improved by adding more LF training simulations. This is likely because the error is dominated by the delta function in L1 at small scales. `MF-Box` achieves an average error at the 1 per cent level with 30L1+30L2 + 3HF, as expected from Section 5.

In summary, we show that the improvement of `MF-Box` happens at small scales ($k > 2\,h\mathrm{Mpc}^{-1}$) at the higher redshift bins ($z \in [1, 2, 3]$). This is primarily because the L1 node at these redshifts has the delta function feature from the initial particle grid dominating on small scales.

[7]As discussed in Ho et al. (2022), our HF power spectra are $\sim 0.1-10$ per cent error compared with EuclidEmulator2.

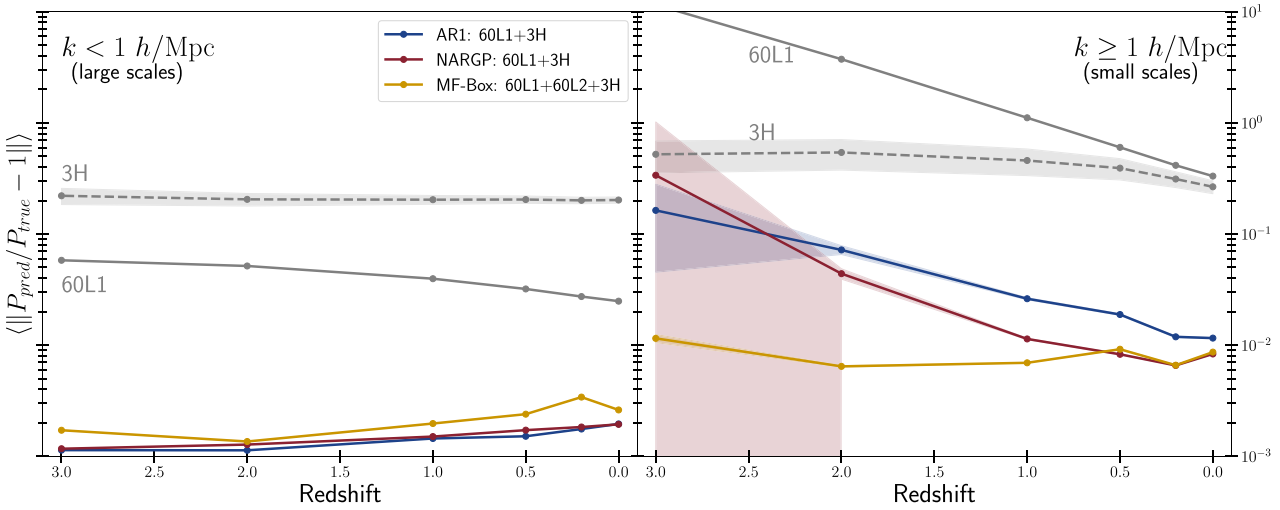

Figure 11. Relative errors averaged over all k modes (split into large and small scales) for different multifidelity models (AR1 (blue), NARGP (red), and `MF-Box` (yellow), broken down into different redshift bins. The grey-dashed line is the HF-only emulator using 3 H simulations, and the solid grey line is the LF-only emulator using 60 L1 simulations. The shaded area is the variance among different test simulations. `MF-Box` improves the emulation at small scales at higher redshifts ($z \geq 1$). We do not include the variance of LFEmu (60L1) because the variance is too large.

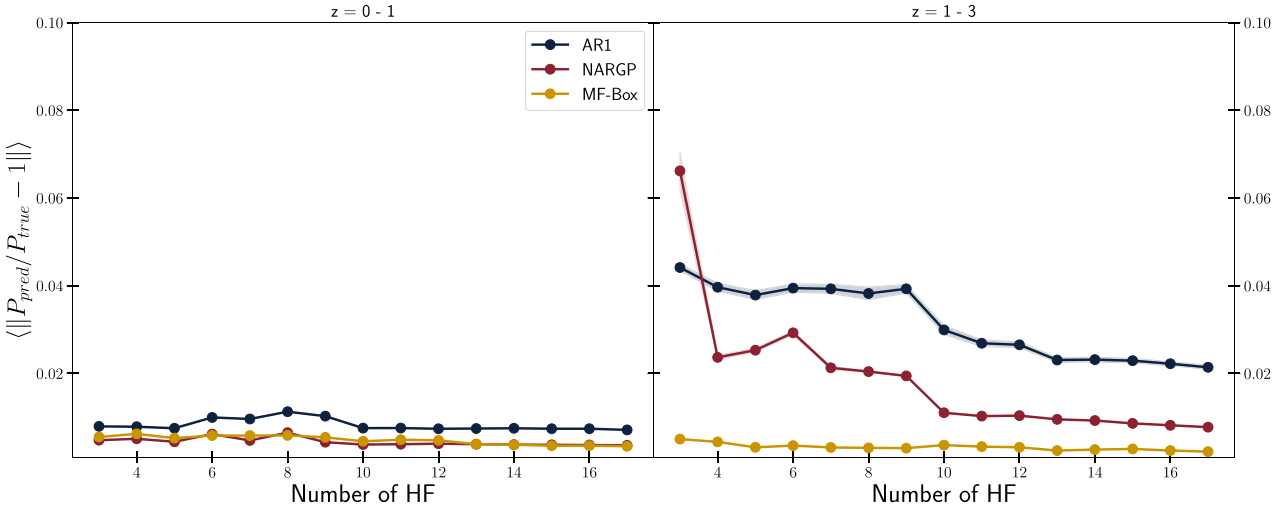

Figure 12. Relative error as a function of the number of HF training points for different multifidelity methods: AR1 (blue), NARGP (red), and `MF-Box` (yellow). The range of the number of HF points is relatively small, so the error estimate trend is unclear. However, in general, the emulation error decreases with more HF points. Left panel: averaged relative error for $z \in [0, 0.2, 0.5]$. Right panel: averaged relative error for $z \in [1, 2, 3]$.

6.2 Emulation with various box sizes

In Section 6.1, we have learned that we can achieve better emulation performance by incorporating an LF node in a smaller box. This section examines how `MF-Box`'s emulation error changed as a function of the L2 box size.

Fig.14 shows the emulation error as a function of L2 box size, averaging over all k bins and splitting into two redshift bins. We include AR1, NARGP, and `MF-Box`. In this section, we use the L2 node as the LF node for both AR1 and NARGP. The left panel shows the error at the low-redshift bin ($z \in [0, 0.2, 0.5]$). AR1 and NARGP have < 1 per cent error with L2 $= 256\,\mathrm{Mpc}\ \ h^{-1}$, but the error gets worse when the L2 box size becomes smaller due to the cosmic variance at large scales. On the other hand, `MF-Box` error stays flat for L2 $\in [100, 224]\,\mathrm{Mpc}\ \ h^{-1}$.

The right panel of Fig. 14 shows the error versus L2 box size at the high-redshift bin, $z \in [1, 3]$. All models show a decrease in error using a smaller L2 box size in training. This is mainly due to the feature at the initial interparticle spacing mentioned in Section 6.1. If a smaller L2 is used, the feature moves to smaller scales, away from those we are emulating, causing a decline of error from the large L2 box to the small L2 box size.

To help visualize the performance change on different scales, we show in Fig. 15 the emulation error as a function of k, averaged over all redshift bins. As Fig. 15 shows, for different L2 sizes, `MF-Box` accuracy only changes at the small scales with $k > 3\,h\mathrm{Mpc}^{-1}$. This is not a surprise because all `MF-Box` models share the same L1 node (128^2 simulations in $256\,\mathrm{Mpc}\ \ h^{-1}$), and thus the emulation at large scales stays the same. The NARGP shown in Fig. 15 uses L2 with $100\,\mathrm{Mpc}\ \ h^{-1}$ as an LF node. Its performance is worse than `MF-Box` with L2 $= 100\,\mathrm{Mpc}\ \ h^{-1}$ at all k bins.

To sum up, the error of `MF-Box` changed as a function of L2 box size: using a smaller L2 can result in better `MF-Box` accuracy. The

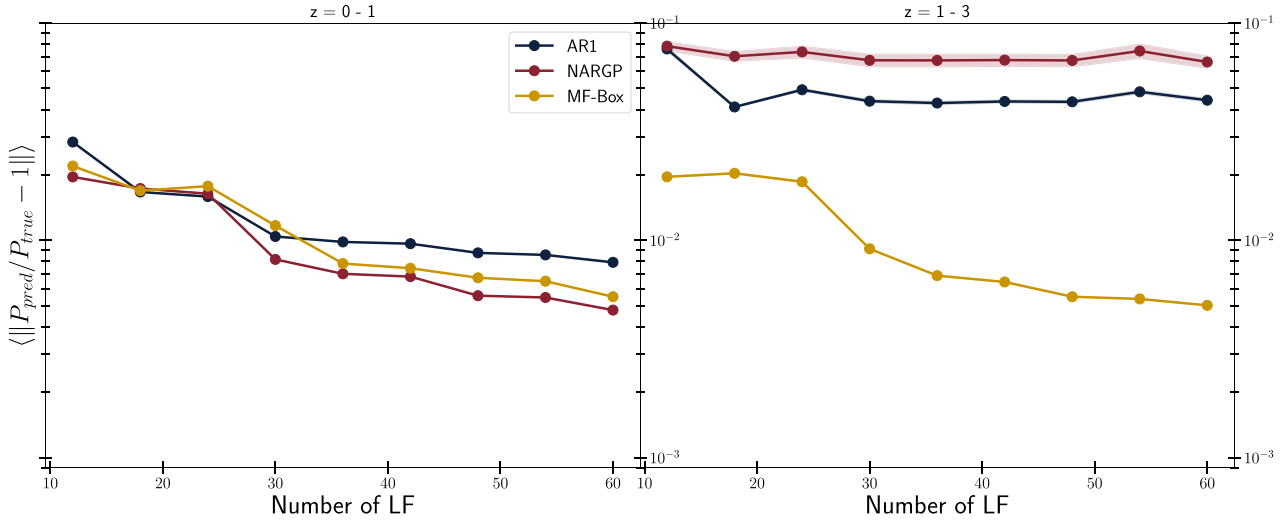

Figure 13. Relative errors for AR1 (blue), NARGP (red), and `MF-Box` (yellow) as a function of LF points, splitting into two redshift bins. Left panel: averaged error for $z \in [0, 0.2, 0.5]$. Right panel: averaged error for $z \in [1, 2, 3]$.

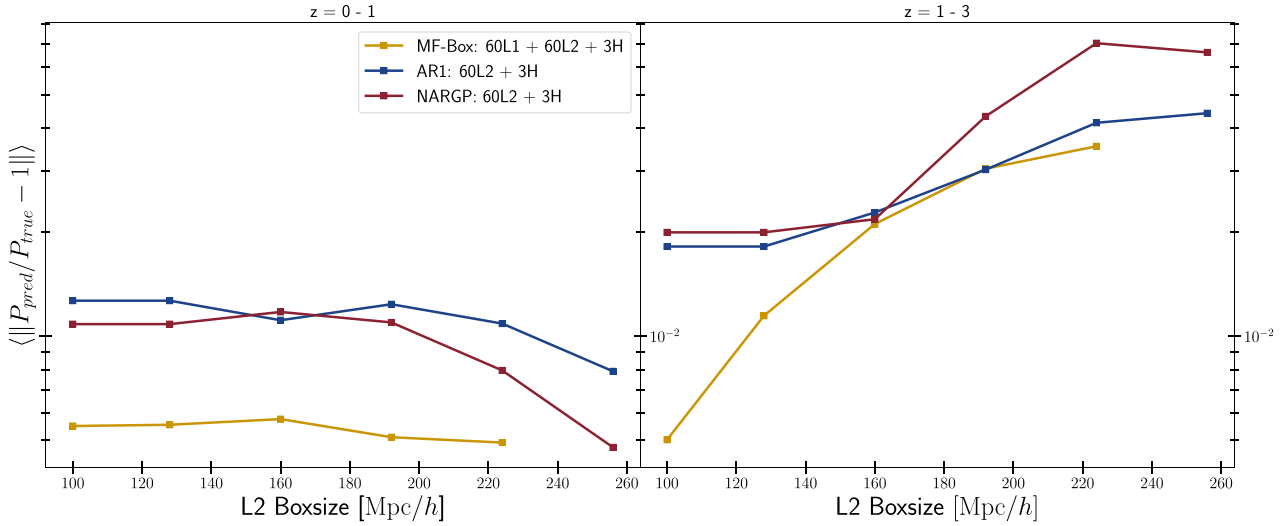

Figure 14. Relative errors of multifidelity emulation as a function of L2 boxsize, for AR1 (blue), NARGP (red), and `MF-Box` (yellow). Note that we use L2 instead of L1 for AR1 and NARGP models.

improvement caused by L2 is mostly at small scales ($k > 2\,h\text{Mpc}^{-1}$) at higher redshift bins ($z = 1, 2, 3$).

6.3 Runtime comparison

We will compare the costs of each method in this section. Fig. 16 shows the error of different emulators as a function of node hours for the training simulations. A similar compute time versus accuracy plot can be found in fig. 4 of Ho et al. (2022), albeit only for $z = 0$. We performed the MP-Gadget simulations at high-performance computing centre (HPCC) at UC Riverside,[8] each compute node has 32 intel Broadwell cores.

To understand Fig. 16, we can start with the HF only emulators ([3–11] HF). This is the emulator we would train before we have multifidelity methods. HF-only emulator shows a steady improvement with an increase in run time. However, the error gradient gets flatter with more training points, indicating the difficulty of improving an emulator at a highly accurate regime.

This trend is intuitive because the error of an emulator roughly scales as a power-law function, (number of training points)$^{-\frac{\nu}{d}}$. Each line in Fig. 16 is a segment of different power-law models. In this view, we can see AR1 and NARGP follow two very similar trends, except one has a lower mean emulation error.

Switching the focus to `MF-Box`, we can see the mean error of the power law is ∼6−8 times better than AR1 and NARGP. The error for both AR1 and NARGP plateaus, implying that adding new simulations will not increase the emulator's accuracy. The only way to improve the emulation at a similarly good efficiency is using small-box simulations through `MF-Box`.

Recall the HF/L1 ratios in Fig. 2. L1 is roughly at ∼ 5 per cent error at large scales. On the other hand, the L2-only emulator is at ∼ 10 per cent error. Using an `MF-Box`, the information carried by L1 and L2 is corrected to be at ∼ 0.5 per cent level, which is a

[8] https://hpcc.ucr.edu

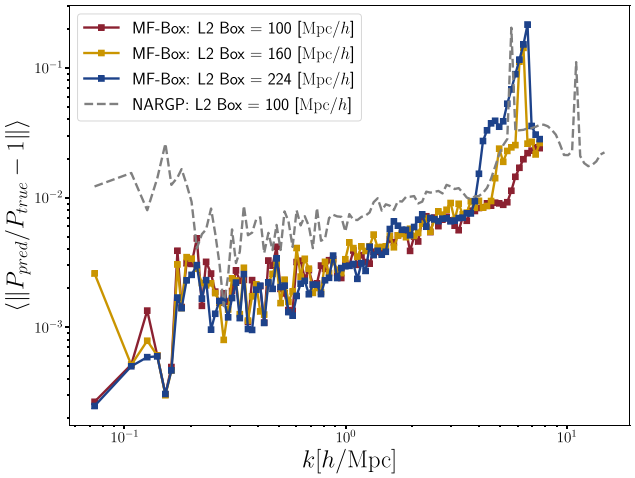

Figure 15. Relative errors averaged over redshift bins, as a function of k modes. `MF-Box` with 224 Mpc h^{-1} L2 (blue), `MF-Box` with 160 Mpc h^{-1} L2 (yellow), and `MF-Box` with 100 Mpc h^{-1} L2 (red). The grey-dashed line is the NARGP model uses 100 Mpc h^{-1} L2.

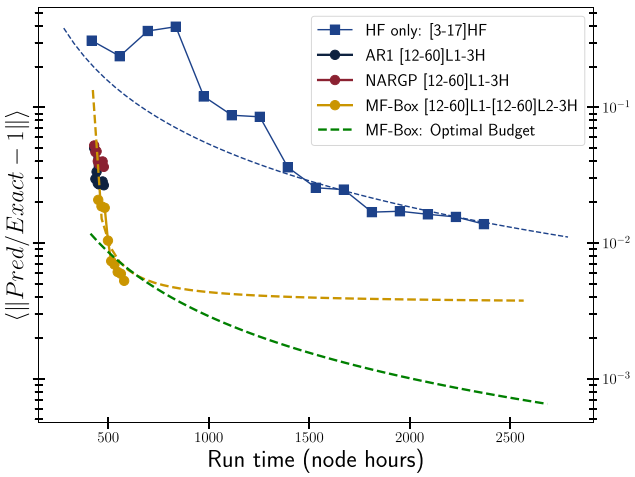

Figure 16. Runtime comparison in node hours. We average the error across redshift bins $z = [0, 0.2, 0.5, 1, 2, 3]$ and average across k bins. AR1 and NARGP perform similarly to `MF-Box` at $z < 1$. Dashed lines are the predicted error based on the error function equation (13), which we inferred in Section 5.

substantial improvement given that only 3 HF simulations are utilized to establish correlations between fidelities.

7 CONCLUSIONS

In this work, we show that our multifidelity emulation, `MF-Box` (model structure refers to Fig. 1, and simulation data refer to Table 1), can combine simulations from different box sizes to achieve improved overall emulator accuracy. `MF-Box` has a higher accuracy improvement per CPU hour than the multifidelity method with only one box size. The framework is adaptable to different simulation suites and emulation problems.

We summarize the key contributions of this work as follows:

(i) Propose a new multifidelity emulation, `MF-Box`, combining information from different simulation box sizes: Using the in-tree graph of GMGP (Ji et al. 2021), we can fuse cheap LF simulations from multiple box sizes in one unified machine-learning model. Simulations in a large box capture large-scale statistics, while the simulations in a small box can improve small-scale statistics. Previously, the cheapest way to improve `MFEmulator` was by increasing the particle load in the LF node, which scales as $\sim \mathcal{O}(N_p^3)$. `MF-Box` opens a new avenue to add additional information to the multifidelity emulation framework in a cheaper way.

(ii) Leverage accurate and systematic-free information from L2 to improve multifidelity emulation accuracy**:** L2 provides unique information absent in L1, and also acts as a cross-check for L1. Systematic errors or unknown bugs in LF nodes can limit the effectiveness of multifidelity methods, as it relies on existing information. Ho et al. (2022) identified such a limitation, noting that systematic errors present in the LF node can make achieving high accuracy difficult. `MF-Box` helps resolve the systematic in one LF node by introducing an additional L2 node without the systematic. It is worth noting that systematic errors may exist in both L1 and L2 nodes, but `MF-Box` can help mitigate these errors by cross-checking the information provided by two nodes, as long as the systematic errors are present at different scales.

(iii) Power-law analysis of emulation errors in multifidelity modelling with `MF-Box`: In Section 5, we present an error analysis of `MF-Box` models. We empirically estimate the emulation error function, which follows a power-law decay with respect to the number of training simulations. This explains why it is difficult to improve single-fidelity emulators which are already per cent-level accurate. Multifidelity emulation shows advantageous in reducing the overall cost and time required to achieve high accuracy. The estimated error function can also serve as a guide for optimizing resource allocation across fidelity nodes, facilitating the development of accurate emulators in a more efficient use of resources.

`MF-Box` also opens up opportunities to experiment with different ways to implement multifidelity emulation in cosmology. The second LF node, L2, can be anything that brings new information to a multifidelity emulator. For example, it could be a node that runs using hydrodynamical simulations, or a node that uses a linear perturbation theory code. One example could be L1 runs with dark-matter only simulations at high resolution, L2 runs with hydrodynamical simulations at low resolution (and in a small box), and an HF node as hydrodynamical simulations at high resolution. This way, the cosmological dependence of the baryonic effects is captured by L2, and L1 gives us highly accurate gravitational clustering. `MF-Box`, using a different box size in an additional LF node, is just a simple example to demonstrate the flexibility of this method.

The main remaining limitation of our multifidelity emulation framework is that the highest fidelity node must be in the training set, and encompass the largest box and highest resolution. In other words, our multifidelity framework cannot extrapolate to predict the results of a simulation with a resolution higher than the HF node.

Future applications of our multifidelity emulation include applying the `MF-Box` to the accurate high-resolution simulations, where the resolution can match the future experiments. We may also apply `MF-Box` to different cosmological probes, especially applying to the beyond two-point statistics, such as weak-lensing peak counts and scattering transform coefficients.

SOFTWARE

We used the `GPy` (GPy 2012) package for Gaussian processes. For multifidelity kernels, we moderately modified the multifidelity

submodule from `emukit` (Paleyes et al. 2019).[9] For the dGMGP model, we used the code provided by Ji et al. (2021), which uses `GPy`. For maxmin SLHD, we use the R software `maximinSLHD` (Ba et al. 2015). We used the MP-GADGET (Feng et al. 2018) software for simulations.[10] We generated customized dark matter-only simulations using Latin hypercubes through a modified version of `SimulationRunner`.[11] Fig. 1 is plotted using `gaepsi2`.[12] We also make use of the following python libraries: `matplotlib` (Hunter 2007), `numpy` (Harris et al. 2020), `scipy` (Virtanen et al. 2020), and `pymc` (Salvatier et al. 2016).

Our code is publicly available at https://github.com/jibanCat/matter_emu_mfbox, including an additional notebook example for the Tensorflow Probability[13] (Dillon et al. 2017) implementation of `MF-Box`.

ACKNOWLEDGEMENTS

We would like to thank Dr Simon Mak and Irene Ji for motivating this work and providing the Python code for the Graphical Model Gaussian Process (GMGP). M-FH thanks Yueying Ni for her guidance in creating visually appealing simulation figures for MP-Gadget using `gaepsi2`. We thank Marjuka Ferdousi Lazin and Harini Venkatesan for their helpful discussions on combining different LF models in `MFEmulator` during the early stages of the project. We acknowledge the Cosmology from Home 2022 conference for providing a valuable virtual platform for discussions. M-FH thanks Sum (Mahdi) Qezlou, Yongda Zhu, Pak Kau Lim, Liz Finney, Reza Monadi, and Archana Aravindan for their valuable feedback on this work during the UCR's Physics and Astronomy Student Seminar (PASS). M-FH thanks Yanhui Yang for the valuable feedback. M-FH acknowledges the support of a National Aeronautics and Space Administration FINESST grant under No. ASTRO20-0022. M-AF is supported by a National Science Foundation Graduate Research Fellowship under grant No. DGE-1326120. SB acknowledges funding from NASA ATP 80NSSC22K1897. Computations were performed using the computer clusters and data storage resources of the HPCC, which were funded by grants from NSF (MRI-2215705, MRI-1429826) and NIH (1S10OD016290-01A1), and Frontera computing project at the Texas Advanced Computing Center (TACC), which was funded through National Science Foundation award OAC-1818253.

DATA AVAILABILITY

The simulation data are available at https://github.com/jibanCat/MFBoxData. The emulator demo codes are available at https://github.com/jibanCat/matter_emu_mfbox.

REFERENCES

Alsing J. et al., 2020, ApJS, 249, 5
Aricò G., Angulo R. E., Zennaro M., 2022, Open Res Europe, 1, 152
Aricò G., Angulo R. E., Contreras S., Ondaro-Mallea L., Pellejero-Ibañez M., Zennaro M., 2021, MNRAS, 506, 4070
Arnold C., Li B., Giblin B., Harnois-Déraps J., Cai Y.-C., 2022, MNRAS, 515, 4161
Auld T., Bridges M., Hobson M. P., Gull S. F., 2007, MNRAS, 376, L11
Auld T., Bridges M., Hobson M. P., 2008, MNRAS, 387, 1575
Ba S., Myers W. R., Brenneman W. A., 2015, Technometrics, 57, 479
Barnes J., Hut P., 1986, Nature, 324, 446
Bevins H. T. J., Handley W. J., Fialkov A., de Lera Acedo E., Javid K., 2021, MNRAS, 508, 2923
Bird S., Rogers K. K., Peiris H. V., Verde L., Font-Ribera A., Pontzen A., 2019, J. Cosmol. Astropart. Phys., 2019, 050
Bird S., Ni Y., Di Matteo T., Croft R., Feng Y., Chen N., 2022, MNRAS, 512, 3703
Bocquet S., Heitmann K., Habib S., Lawrence E., Uram T., Frontiere N., Pope A., Finkel H., 2020, ApJ, 901, 5
Boruah S. S., Eifler T., Miranda V., M S. K. P., 2022, MNRAS, 518, 4818
Bye C. H., Portillo S. K. N., Fialkov A., 2022, ApJ, 930, 79
Cabayol-Garcia L., Chaves-Montero J., Font-Ribera A., Pedersen C., 2023, MNRAS, 525, 3499
Chartier N., Wandelt B. D., 2022, MNRAS, 515, 1296
Chartier N., Wandelt B., Akrami Y., Villaescusa-Navarro F., 2021, MNRAS, 503, 1897
Cheung D. H. T., Wong K. W. K., Hannuksela O. A., Li T. G. F., Ho S., 2021, Phys. Rev. D, 106, 083014
Christen J. A., Fox C., 2005, J. Comput. Graph. Statist., 14, 795
Cohen A., Fialkov A., Barkana R., Monsalve R. A., 2020, MNRAS, 495, 4845
Cole A., Miller B. K., Witte S. J., Cai M. X., Grootes M. W., Nattino F., Weniger C., 2022, J. Cosmol. Astropart. Phys., 2022, 004
Currin C., Mitchell T., Morris M., Ylvisaker D., 1991, J. Am. Statist. Assoc., 86, 953
Damianou A., Lawrence N., 2013, in Carvalho C. M., Ravikumar P.eds, Proceedings of Machine Learning Research Vol. 31, Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics. PMLR, Scottsdale, Arizona, AZ, p. 207,
Davies C. T., Cautun M., Giblin B., Li B., Harnois-Déraps J., Cai Y.-C., 2021, MNRAS, 507, 2267
Dillon J. V. et al., 2017, preprint (arXiv:1711.10604)
El Gammal J., Schöneberg N., Torrado J., Fidler C., 2022, preprint (arXiv:2211.02045)
Euclid Collaboration et al., 2019, MNRAS, 484, 5509
Euclid Collaboration et al., 2021, MNRAS, 505, 2840
Feng Y., Bird S., Anderson L., Font-Ribera A., Pedersen C., 2018, MP-Gadget/MP-Gadget: A tag for getting a DOI, https://doi.org/10.5281/zenodo.1451799
Fernandez M. A., Ho M.-F., Bird S., 2022, MNRAS, 517, 3200
GPy, since 2012, GPy: A Gaussian process framework in python, http://github.com/SheffieldML/GPy
Giblin B., Cataneo M., Moews B., Heymans C., 2019, MNRAS, 490, 4826
Giblin B., Cai Y.-C., Harnois-Déraps J., 2023, MNRAS, 520, 1721
Giri S. K., Schneider A., 2021, J. Cosmol. Astropart. Phys., 2021, 046
Günther S., Lesgourgues J., Samaras G., Schöneberg N., Stadtmann F., Fidler C., Torrado J., 2022, J. Cosmol. Astropart. Phys., 2022, 63
Hand N., Feng Y., Beutler F., Li Y., Modi C., Seljak U., Slepian Z., 2018, AJ, 156, 160
Harnois-Déraps J., Giblin B., Joachimi B., 2019, A&A, 631, A160
Harnois-Déraps J., Hernandez-Aguayo C., Cuesta-Lazaro C., Arnold C., Li B., Davies C. T., Cai Y.-C., 2022, MNRAS, 525, 6336
Harris C. R. et al., 2020, Nature, 585, 357
Heitmann K., Higdon D., White M., Habib S., Williams B. J., Lawrence E., Wagner C., 2009, ApJ, 705, 156
Heitmann K., White M., Wagner C., Habib S., Higdon D., 2010, ApJ, 715, 104
Heitmann K., Lawrence E., Kwan J., Habib S., Higdon D., 2014, ApJ, 780, 111
Ho M.-F., Bird S., Shelton C. R., 2022, MNRAS, 509, 2551
Holdship J., Viti S., Haworth T. J., Ilee J. D., 2021, A&A, 653, A76
Hunter J. D., 2007, Comput. Sci. Eng., 9, 90
Ji Y., Mak S., Soeder D., Paquet J.-F., Bass S. A., 2021, preprint (arXiv:2108.00306)

[9] https://github.com/EmuKit/emukit
[10] https://github.com/MP-Gadget/MP-Gadget
[11] Original: https://github.com/sbird/SimulationRunner; modified: https://github.com/jibancat/SimulationRunnerDM
[12] https://github.com/rainwoodman/gaepsi2
[13] https://www.tensorflow.org/probability

Ji Y., Shaowu Yuchi H., Soeder D., Paquet J. F., Bass S. A., Roshan Joseph V., Wu C. F. J., Mak S., 2022, preprint (arXiv:2209.13748)
Jo Y. et al., 2023, ApJ, 944, 67
Kennedy M., O'Hagan A., 2000, Biometrika, 87, 1
Kern N. S., Liu A., Parsons A. R., Mesinger A., Greig B., 2017, ApJ, 848, 23
Kodi Ramanah D., Charnock T., Villaescusa-Navarro F., Wandelt B. D., 2020, MNRAS, 495, 4227
Kugel R. et al., 2023, preprint (arXiv:2306.05492)
Lam R., Allaire D., Willcox K. E., 2015, 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, https://doi.org/10.2514/6.2015-0143
Lawrence E. et al., 2017, ApJ, 847, 50
Leclercq F., 2018, Phys. Rev. D, 98, 063511
Lesgourgues J., 2011, preprint (arXiv:1104.2932)
Li Y., Ni Y., Croft R. A. C., Di Matteo T., Bird S., Feng Y., 2021, Proc. Natl. Acad. Sci, 118, e2022038118
Lykkegaard M. B., Mingas G., Scheichl R., Fox C., Dodwell T. J., 2020, preprint (arXiv:2012.05668)
McClintock T. et al., 2019, ApJ, 872, 53
Moran K. R. et al., 2023, MNRAS, 520, 3443
Neveux R. et al., 2022, MNRAS, 516, 1910
Ni Y., Li Y., Lachance P., Croft R. A. C., Di Matteo T., Bird S., Feng Y., 2021, MNRAS, 507, 1021
Ni Y. et al., 2022, MNRAS, 513, 670
Nishimichi T. et al., 2019, ApJ, 884, 29
Nygaard A., Brinch Holm E., Hannestad S., Tram T., 2022, J. Cosmol. Astropart. Phys., 2023, 29
O'Hagan A., 2006, Reliab. Eng. Syst. Saf., 91, 1290
Paleyes A., Pullin M., Mahsereci M., Lawrence N., González J., 2019, in Second Workshop on Machine Learning and the Physical Sciences, NIPS.
Pedersen C., Font-Ribera A., Rogers K. K., McDonald P., Peiris H. V., Pontzen A., Slosar A., 2021, J. Cosmol. Astropart. Phys., 2021, 033
Peherstorfer B., Willcox K., Gunzburger M., 2018, SIAM Rev., 60, 550
Pellejero-Ibañez M., Angulo R. E., Aricó G., Zennaro M., Contreras S., Stücker J., 2020, MNRAS, 499, 5257
Perdikaris P., Raissi M., Damianou A., Lawrence N. D., Karniadakis G. E., 2017, Proc. R. Soc. A., 473, 20160751
Poloczek M., Wang J., Frazier P. I., 2017, Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., preprint (arXiv:1603.00389)
Qian P. Z. G., 2012, J. Am. Statist. Assoc., 107, 393
Rasmussen C. E., Williams C. K. I., 2005, Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning), The MIT Press, Cambridge, MA, United States.
Rogers K. K., Peiris H. V., 2021a, Phys. Rev. D, 103, 043526
Rogers K. K., Peiris H. V., 2021b, Phys. Rev. Lett., 126, 071302
Rogers K. K., Peiris H. V., Pontzen A., Bird S., Verde L., Font-Ribera A., 2019, J. Cosmol. Astropart. Phys., 2019, 031
Rogers J. G., Janó Muñoz C., Owen J. E., Booth R. A., 2021, MNRAS, 519, 6028
Salcido J., McCarthy I. G., Kwan J., Upadhye A., Font A. S., 2023, MNRAS, 523, 2247
Salvatier J., Wiecki T. V., Fonnesbeck C., 2016, PeerJ Comput. Sci, 2, e55
Santner T. J., Williams B. J., Notz W. I., 2003, The Design and Analysis of Computer Experiments. Springer series in statistics, Springer, Berlin/Heidelberg, Germany
Schneider A. et al., 2016, J. Cosmol. Astropart. Phys., 2016, 047
Springel V., Hernquist L., 2003, MNRAS, 339, 289
Springel V., Pakmor R., Zier O., Reinecke M., 2021, MNRAS, 506, 2871
Spurio Mancini A., Piras D., Alsing J., Joachimi B., Hobson M. P., 2022, MNRAS, 511, 1771
Takhtaganov T., Lukić Z., Müller J., Morozov D., 2021, ApJ, 906, 74
Vernon I., Liu J., Goldstein M., Rowe J., Topping J., Lindsey K., 2018, BMC Syst. Biol., 12, 1
Villaescusa-Navarro F. et al., 2021, ApJ, 915, 71
Virtanen P. et al., 2020, Nat. Methods, 17, 261
Wendland H., 2004, Scattered Data Approximation. Cambridge Monographs on Applied and Computational Mathematics, Cambridge Univ. Press, Cambridge
Zel'Dovich Y. B., 1970, A&A, 500, 13
Zürcher D. et al., 2022, MNRAS, 511, 2075

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