A&A, 685, A156 (2024)
https://doi.org/10.1051/0004-6361/202347892
© The Authors 2024

Astronomy & Astrophysics

Sesame: A power spectrum emulator pipeline for beyond-ΛCDM models

Renate Mauland, Hans A. Winther, and Cheng-Zong Ruan

Institute of Theoretical Astrophysics, University of Oslo, PO Box 1029 Blindern, 0315 Oslo, Norway
e-mail: renate.mauland-hus@astro.uio.no

Received 6 September 2023 / Accepted 2 March 2024

ABSTRACT

The mysterious nature of the dark sector of the Λ-cold-dark-matter (ΛCDM) model is one of the main motivators behind the study of alternative cosmological models. A central quantity of interest for these models is the matter power spectrum, which quantifies structure formation on various scales and can be cross-validated through theory, simulations, and observations. Here, we present a tool that can be used to create emulators for the non-linear matter power spectrum, and similar global clustering statistics, for models beyond ΛCDM with very little computation effort and without the need for supercomputers. We use fast approximate N-body simulations to emulate the boost, $B(k,z) = P_{\rm beyond-\Lambda CDM}(k,z)/P_{\Lambda\rm CDM}(k,z)$, and then rely on existing high-quality emulators made for ΛCDM to isolate $P_{\rm beyond-\Lambda CDM}(k,z)$. Since both the ΛCDM and beyond-ΛCDM models are simulated in our approach, some of the lack of power on small scales due to the low force-resolution in the simulations is factored out, allowing us to extend the emulator to $k \sim 3$–$5\,h\,{\rm Mpc}^{-1}$ and still maintain good accuracy. In addition, errors from the simulation and emulation process can easily be estimated and factored into the covariance when using the emulator on data. As an example of using the pipeline, we create an emulator for the well-studied $f(R)$ model with massive neutrinos, using approximately 3000 CPU hours of computation time. Provided with the paper is a fully functioning pipeline that generates parameter samples, runs a Boltzmann solver to produce initial conditions, runs the simulations, and then gathers all the data and runs it through a machine learning module to develop the emulator. This tool, named Sesame, can be used by anyone to generate a power spectrum emulator for the cosmological model of their choice.

Key words. gravitation – neutrinos – methods: numerical – methods: statistical – large-scale structure of Universe

1. Introduction

The Λ-cold-dark-matter (ΛCDM) model describes our Universe well, yet two of its main components remain elusive. The true natures of dark matter and dark energy are still unknown, but their impact on the Universe has been, and continues to be, widely studied across multiple research fields. In an attempt to forego the dark energy component of the ΛCDM model, alternative theories of gravity have become a popular avenue to explore. These beyond-ΛCDM models (see e.g. Clifton et al. 2012; Koyama 2016; Wright et al. 2023) have an effect on structure formation, leaving an imprint on the matter power spectrum. This can be further studied through the use of cosmological simulations, which typically require a large amount of computing resources for high-resolution simulations capable of accurately distinguishing between models down to small scales. In addition, a simulation is only performed for a specified set of cosmological parameters, requiring a rerun for any parameter changes. To forgo both of these issues, emulators can be created for desired statistical observables, like the matter power spectrum - a key observable whose theoretical prediction is needed to constrain beyond-ΛCDM models in current and near-future weak-lensing surveys (J-PAS Collaboration 2014; LSST Collaboration 2019; DES Collaboration 2021; Euclid Collaboration 2022). The emulators (Heitmann et al. 2013; Kwan et al. 2015; Giblin et al. 2019; Nishimichi et al. 2019; Angulo et al. 2021; Euclid Collaboration 2021; Moran et al. 2023) are typically constructed by performing a high number of N-body simulations within some parameter space, and then interpolating to access any desired parameter value. This can be done, for example, through the use of machine learning, training a neural network on highly accurate simulation data. As mentioned above, the simulation step can be computationally expensive, but once it is performed and the following training is done, the emulators are simple to use and have both minimal time and memory requirements.

Although highly accurate, a limit of this approach is the ability to easily extend the resulting emulator to new cosmological models. In this paper, we present a full pipeline using fast approximate N-body simulations and neural network training to create an emulator for the matter power spectrum boost, $B(k,z) = P_{\rm beyond-\Lambda CDM}(k,z)/P_{\Lambda\rm CDM}(k,z)$. The approximate simulations employ the comoving Lagrangian acceleration (COLA) method (Tassev et al. 2013) to simulate both the ΛCDM and beyond-ΛCDM models (Valogiannis & Bean 2017; Winther et al. 2017; Wright et al. 2017, 2023; Brando et al. 2022; Fiorini et al. 2022; Brando et al. 2023), allowing us to extract the boost up to scales of $k \sim 3-5\,h\,{\rm Mpc}^{-1}$ to a few percent accuracy. The pipeline is named Sesame – from simulations to emulators using approximate methods. As a demonstration of Sesame, we create an emulator for the boost between the Hu-Sawicki $f(R)$ model (Hu & Sawicki 2007) and a dynamical dark energy model, $w_0w_{\rm a}$CDM. In $f(R)$-modified gravity, an additional function of the Ricci scalar, R, is added to the general relativity (GR) framework (Buchdahl 1970). This function can be designed to recreate a similar expansion history as ΛCDM, without the need for dark energy. Still, as the nature of gravity is modified, resulting observational signals are expected (see e.g. de Felice & Tsujikawa

2010, for a detailed review). One such signal is the enhancement of structure formation on scales smaller than the Compton wavelength of the scalaron – the scalar degree of freedom of the $f(R)$ theory, $\mathrm{d}f/\mathrm{d}R$ (e.g. Hu & Sawicki 2007; Pogosian & Silvestri 2008; Cataneo et al. 2015). This shows up in the matter power spectrum.

In addition to exploring universe models besides ΛCDM, calculations and simulations within the ΛCDM framework are continuously expanded to reach higher levels of accuracy. One such extension is the inclusion of massive neutrinos. These lightweight particles have often been excluded from cosmological simulations due to their low impact compared to cold dark matter (cdm), which makes up about 84% (Planck Collaboration VI 2020) of the matter content of the Universe. However, improvements in telescopes and satellites now give us an observational accuracy high enough to measure the impact of neutrinos on structure formation – suppression on scales smaller than the neutrino free-streaming length (Lesgourgues & Pastor 2006). Surveys like the newly launched *Euclid* satellite (Laureijs et al. 2011) and the ongoing DESI (Dark Energy Spectroscopic Instrument) experiment[1] (DESI Collaboration 2016) can measure the effect of massive neutrinos and thereby put tighter constraints on the neutrino mass scale. Because of this, we include modified gravity, massive neutrinos, and dark energy in the form of the well-known w_0w_a Chevallier-Polarski-Linder (CPL) parametrisation (Chevallier & Polarski 2001; Linder 2003) when creating our emulator. The inclusion of massive neutrinos in the $f(R)$ simulations is also particularly important, due to the degeneracy between the effects of neutrinos and $f(R)$-modified gravity on structure formation on non-linear scales (e.g. Baldi et al. 2014).

Simulations including massive neutrinos (Potter et al. 2017; Adamek et al. 2017; Liu et al. 2018; Dakin et al. 2019; Partmann et al. 2020; Weinberger et al. 2020; Springel et al. 2021; Euclid Consortium 2023), modified gravity (Li et al. 2012; Puchwein et al. 2013; Llinares et al. 2014; Winther et al. 2015; Hassani & Lombriser 2020; Ruan et al. 2022; Wright et al. 2023), and both (Baldi et al. 2014; Wright et al. 2017; Giocoli et al. 2018; Mauland et al. 2023) already exist with various methods of implementation, along with models, fits, and emulators to extract the boost or the matter power spectrum directly for these cosmological models (e.g. Zhao 2014; Winther et al. 2019; Hannestad et al. 2020; Bose et al. 2020, 2021, 2023; Ramachandra et al. 2021; Euclid Collaboration 2021; Arnold et al. 2022; Gupta et al. 2023; Moran et al. 2023). The main takeaway from this paper is therefore not the $f(R)$-modified gravity emulator (although it will be provided), but the full pipeline, Sesame, which includes the drawing of parameter samples, running the simulations, training the neural network, and creating the emulator for the boost, $B(k,z)$. This tool can be used to produce an emulator for a desired cosmological model by implementing said model into the simulations and using a suitable Boltzmann solver to extract the initial conditions. The resulting accuracy of both the simulations and the emulator can be tuned by the choice of simulation settings and neural network architecture.

This paper is structured as follows: In Sect. 2, we present some background theory for the matter power spectrum, $f(R)$-modified gravity, and massive neutrinos. This is followed by an outline of the methods applied in Sect. 3, including a description of the full pipeline. In Sect. 4 we go through some simulation details, and then present our results in Sect. 5. Finally, we conclude in Sect. 6.

[1] https://www.desi.lbl.gov

2. Theory

In this section, we present some background information for the key components of this work. We first outline the necessary details on the matter power spectrum, followed by $f(R)$-modified gravity and massive neutrinos.

2.1. Matter power spectrum

The matter power spectrum, $P(k)$, is defined as (e.g. Peebles 1980; Dodelson & Schmidt 2020):

$$(2\pi)^3 P(k)\delta_\mathrm{D}(\boldsymbol{k}-\boldsymbol{k}') = \langle\tilde{\delta}(\boldsymbol{k})\tilde{\delta}(\boldsymbol{k}')^*\rangle, \tag{1}$$

where k is the wavenumber, δ_D is the Dirac-delta function, and $\tilde{\delta}(\boldsymbol{k})$ is the Fourier transform of the overdensity field, $\delta(\boldsymbol{x})$. The power spectrum is the Fourier transform of the two-point correlation function, $\xi(\boldsymbol{r})$, which describes the excess probability, over random, of finding two objects separated by a distance $\boldsymbol{r}$. Analysing the matter power spectrum gives great insight into the clustering of matter at different times and scales, in addition to how variations in cosmological parameters affect structure formation.

When studying alternative models to the concordance ΛCDM model of our Universe, the ratio between the power spectrum in the alternative model and that of ΛCDM holds valuable information about the deviations between them. Different components of a cosmological model, like massive neutrinos or modified gravity, have theoretically predicted impacts on the power spectrum (e.g. Lesgourgues & Pastor 2006; Song et al. 2007; Koyama et al. 2009). As the matter power spectrum can be observed from various surveys (Chabanier et al. 2019; LSST Collaboration 2019; Euclid Collaboration 2022), its shape is well known, and it can therefore be used to constrain these cosmological models. As an example in this paper, we are interested in the differences in the power spectrum between a w_0w_aCDM universe with GR as the gravity model and one with $f(R)$-modified gravity as the gravity model, both with the inclusion of massive neutrinos,

$$B(k,z) = \frac{P_{\mathrm{f(R)}}(k,z \mid \Omega_\Lambda, \Omega_\mathrm{cdm}, \Omega_\mathrm{b}, n_\mathrm{s}, \sigma_8^{\mathrm{f(R)}}, w_0, w_\mathrm{a}, h, M_\nu, f_{\mathrm{R0}})}{P_\mathrm{GR}(k,z \mid \Omega_\Lambda, \Omega_\mathrm{cdm}, \Omega_\mathrm{b}, n_\mathrm{s}, \sigma_8, w_0, w_\mathrm{a}, h, M_\nu)}. \tag{2}$$

Here, Ω_Λ, Ω_CDM, and Ω_b are the energy densities of dark energy, dark matter, and baryons respectively; n_s is the scalar spectral index; h is the Hubble constant today; σ_8 and $\sigma_8^{\mathrm{f(R)}}$ denote the normalisation of the linear matter power-spectra at $z = 0$; f_{R0} is the Hu-Sawicki $f(R)$-modified gravity parameter (see Sect. 2.2); M_ν denotes the sum of the neutrino masses, and w_0 and w_a are dynamical dark energy parameters for the CPL parametrisation of the dark energy equation of state (Chevallier & Polarski 2001; Linder 2003),

$$w = w_0 + w_\mathrm{a}\frac{z}{1+z}, \tag{3}$$

where $w_0 = -1$ and $w_\mathrm{a} = 0$ correspond to a cosmological constant.

2.2. Beyond-ΛCDM models

A vast number of beyond-ΛCDM models are proposed in the literature, and not all of them can be covered here. For a review, we therefore refer the reader to Bull et al. (2016).

The simplest models are dark energy models that mainly modify the background evolution through the Hubble function, $H(a)$. These are the so-called quintessence models (Wetterich 1988) and parametrised models for the dark energy equation of state, $w(a)$, like CPL. Next in the level of complexity, we have models where the quintessence field is coupled to matter (often only dark matter), dubbed coupled-quintessence models (Amendola 2000). Then we have modified gravity models, where an extra degree of freedom is introduced, giving rise to a fifth force for the full matter sector. To be able to evade local gravity constraints, these models often need a screening mechanism to hide the modifications in high-density environments where such gravity tests have been performed (see e.g. Khoury & Weltman 2004a; Clifton et al. 2012; Koyama 2016). In addition to the models mentioned so far, we also have models of dark matter beyond cold dark matter (e.g. axions Marsh 2016), non-standard inflationary models (Martin et al. 2014), and many more. The model we use here for demonstrating how an emulator can be created using Sesame is a $f(R)$ modified gravity model. This is chosen as it is well known and because it is already implemented in the applied code base.

$f(R)$-modified gravity

In $f(R)$-modified gravity theory (Sotiriou & Faraoni 2010), the Einstein-Hilbert action of GR is extended by a function, $f(R)$,

$$S = \left(\int \frac{R + f(R)}{16\pi G} + \mathcal{L}_\mathrm{m}\right)\sqrt{-g}\mathrm{d}^4x. \tag{4}$$

Here, R is the Ricci scalar, G is the Newtonian gravitational constant, $\mathcal{L}_\mathrm{m}$ is the matter Lagrangian density, and g is the determinant of the metric tensor, $g_{\mu\nu}$. The $f(R)$ function can take many forms, one of which is given by

$$f(R) = -m^2\frac{c_1(R/m^2)^n}{c_2(R/m^2)^n + 1}, \tag{5}$$

proposed by Hu & Sawicki (2007). Here c_1, c_2, and n are dimensionless, constant, and non-negative parameters of the model and $m^2 = H_0^2\Omega_\mathrm{cdm}$, with H_0 the value of the Hubble parameter today. This $f(R)$ function was designed so that cosmological tests at high redshifts yield the same results as for GR. In addition, in the limit where $c_2(R/m^2)^n \gg 1$, Eq. (5) can be written as $f(R) = -m^2c_1/c_2 + O((m^2/R)^n)$, showing that a cosmological constant, and thereby a similar background evolution to that of ΛCDM, can be obtained by equating $-m^2c_1/c_2$ with -2Λ. This corresponds to a relation given by $c_1/c_2 = 6\Omega_\Lambda/\Omega_\mathrm{cdm}$ between the two parameters c_1 and c_2. The equation of motion of the scalar degree of freedom, f_R, of the $f(R)$-model is then given by

$$f_\mathrm{R} \equiv \frac{\mathrm{d}f(R)}{\mathrm{d}R} \approx -n\frac{c_1}{c_2}\left(\frac{m^2}{R}\right)^{n+1}. \tag{6}$$

By fixing the value of f_R0, the present-day background value of the scalar degree of freedom, an independent connection can be found for c_1 and c_2. This enables the model to be fully specified by the parameters f_R0 and n. We apply $n = 1$ in this paper.

From theory and simulations, the impact of this form of $f(R)$-modified gravity on structure formation, and thereby the matter power spectrum, can be predicted for various values of f_R0. In general, this modification to gravity enhances structure formation on small scales (Hu & Sawicki 2007; Pogosian & Silvestri 2008; Cataneo et al. 2015), as a result of an attractive force, dubbed the fifth force, which appears in addition to Newtonian gravity. The effects of this, in order for the theory to be compatible with observations (Will 2014), are suppressed in high-density regions due to a chameleon screening effect (Khoury & Weltman 2004b; Brax et al. 2008). The value of f_R0 controls the threshold at which the screening kicks in and recovers GR. Values above $f_\mathrm{R0} \sim -10^{-5}$ are in general ruled out from cosmological observations (Cataneo et al. 2015; Koyama 2016), although massive neutrinos, which have the opposite effect on structure formation, have not always been taken into account in these analyses (Baldi et al. 2014).

2.3. Massive neutrinos

From particle physics, we know that there are three neutrino mass states, ν_i with $i = 1, 2, 3$ (e.g. Thomson 2013). The absolute mass scale, m_{ν_i} (often shortened to m_i), of each state is unknown, but neutrino oscillation experiments give us constraints on the mass difference between the states (Particle Data Group 2022)

$$\begin{aligned}
\Delta m_{21}^2 &= (7.53 \pm 0.18) \times 10^{-5}\,\mathrm{eV}^2,\\
\Delta m_{32}^2 &= (-2.519 \pm 0.033) \times 10^{-3}\,\mathrm{eV}^2\ (\mathrm{IH}),\\
\Delta m_{32}^2 &= (2.437 \pm 0.033) \times 10^{-3}\,\mathrm{eV}^2\ (\mathrm{NH}),
\end{aligned} \tag{7}$$

where IH denotes the inverted hierarchy ($m_3 \ll m_1 < m_2$) and NH the normal hierarchy ($m_1 < m_2 \ll m_3$). This gives a lower bound of $\sum m_\nu \gtrsim 0.1$ eV and $\sum m_\nu \gtrsim 0.06$ eV for the sum of the neutrino masses for the inverted and normal hierarchies respectively. An upper bound is given by $\sum m_\nu \lesssim 2.4$ eV, based on the KATRIN single β-decay experiment (KATRIN Collaboration 2022).

In addition to particle physics experiments, the sum of the neutrino masses can be constrained through cosmological observations. As neutrinos make up a fraction of the energy content of the Universe, given by (Lesgourgues & Pastor 2006)

$$\Omega_\nu \approx \frac{\sum m_\nu}{93.14\,\mathrm{eV}\,h^2}, \tag{8}$$

they affect the formation of structure. At early times, the massive neutrinos are relativistic, and free-stream out of overdense regions. This, in addition to alterations of the background evolution, like the time of matter-radiation equality, leads to a suppression of the matter power spectrum on scales smaller than the neutrino free-streaming length (Lesgourgues & Pastor 2006),

$$\lambda_\mathrm{FS} = 7.7\frac{1+z}{\sqrt{\Omega_\Lambda + \Omega_\mathrm{m}(1+z)^3}}\left(\frac{1\,\mathrm{eV}}{\sum m_\nu}\right)h^{-1}\,\mathrm{Mpc}. \tag{9}$$

Here, $\Omega_\mathrm{m} = \Omega_\mathrm{cdm} + \Omega_\mathrm{b} + \Omega_\nu$ is the total energy density of matter and the other parameters are as explained before. The suppression of structure formation is observable and can help constrain the sum of the neutrino masses. A recent combination of various probes finds $\sum m_\nu \lesssim 0.09$ eV at 95% confidence (Di Valentino et al. 2021) and one of the science goals of the *Euclid* mission is to measure $\sum m_\nu$ to more than 0.03 eV precision through the use of weak gravitational lensing and galaxy clustering (Laureijs et al. 2011).

Although cosmological observations can be used to obtain tighter upper bounds on the sum of the neutrino masses, it is important to take into account the dependence on the choice of a cosmological model. Hu-sawicki $f(R)$-modified gravity, as mentioned above, has the opposite effect of massive neutrinos on structure formation, and thus results in degenerate observables like the matter power spectrum, halo mass function (HMF), halo bias, and void-galaxy cross-correlation function (Baldi et al. 2014; Mauland et al. 2023).

3. Method

In this section, we introduce the methods behind the simulations and machine learning codes used to create the emulator. We also detail the steps that need to be taken before applying the pipeline and the steps taken within the pipeline itself.

3.1. Simulations

The simulations in this paper were performed with the `COLASolver` implemented in the `FML` library[2]. This is a fast and approximate particle-mesh (PM) N-body code which employs the COLA method introduced by Tassev et al. (2013). The `COLASolver` succeeds the `MG-PICOLA`[3] code (Winther et al. 2017) and has various options for cosmologies and gravity models, including dynamical dark energy and $f(R)$-modified gravity. It also contains massive neutrinos, using a grid-based method as proposed in Brandbyge & Hannestad (2009), which is implemented and tested in Wright et al. (2017).

3.1.1. The COLA method

The COLA method (Tassev et al. 2013) is based on the fact that structure formation on large scales is well described by Lagrangian perturbation theory (LPT). We can use this to our advantage and solve for the displacement, $\delta \boldsymbol{x}$, between a particle's LPT trajectory, $\boldsymbol{x}_{\rm LPT}$, and its full trajectory, $\boldsymbol{x}$. The geodesic equation for the particles is given by

$$\frac{{\rm d}\boldsymbol{x}}{{\rm d}\tau} = \boldsymbol{v}, \tag{10}$$

$$\frac{{\rm d}\boldsymbol{v}}{{\rm d}\tau} = -\nabla\Phi, \tag{11}$$

which, when setting $\boldsymbol{x} = \delta\boldsymbol{x} + \boldsymbol{x}_{\rm LPT}$, becomes

$$\frac{{\rm d}\delta\boldsymbol{x}}{{\rm d}\tau} = \delta\boldsymbol{v}, \tag{12}$$

$$\frac{{\rm d}\delta\boldsymbol{v}}{{\rm d}\tau} = -\nabla\Phi - \frac{{\rm d}^2\boldsymbol{x}_{\rm LPT}}{{\rm d}\tau^2}. \tag{13}$$

The additional COLA force is easily computed from the displacement fields that are already calculated when creating the initial conditions. In this COLA frame (the frame co-moving with the LPT trajectories), the initial velocity of the particles is simply $\delta\boldsymbol{v} = 0$, and stays small on large scales during the evolution. This allows us to take much larger time steps than in usual N-body simulations, while still maintaining high accuracy on the largest scales, reducing the simulation time substantially. When increasing the number of timesteps, the method converges towards a full PM N-body code. The COLA method has become an increasingly popular method for cheaply generating simulations and mock galaxy catalogues (Tassev et al. 2015; Feng et al. 2016; Izard et al. 2016; Koda et al. 2016; Leclercq et al. 2020; Brando et al. 2023; Wright et al. 2023).

3.1.2. Screened modified gravity

The `COLASolver` we use already contains implementations of a wide range of modified gravity models, like $f(R)$ gravity, the symmetron, DGP, and Jordan-Brans-Dicke (de Felice & Tsujikawa 2010; Hinterbichler et al. 2011; Dvali et al. 2000; Joudaki et al. 2022). A typical modified gravity model has a Poisson equation which in linear perturbation theory, and in Fourier space, reads (see e.g. Winther et al. 2017)

$$\Phi(k,z) = -\frac{3}{2k^2}\Omega_{\rm m} a\delta_{\rm m}(k,z)\frac{G_{\rm eff}(k,z)}{G}. \tag{14}$$

Here, $G_{\rm eff}(k,z)/G$ represents an effective Newtons constant, which might depend on both time and scale. For example, for the $f(R)$ model, we have

$$\frac{G_{\rm eff}(k,z)}{G} = 1 + \frac{1}{3}\frac{k^2}{k^2 + a^2 m_{\rm f(R)}^2}, \tag{15}$$

where $m_{\rm f(R)}^{-1}$ is the range of the fifth-force. This is exact on linear scales, but it does not include the important screening effect seen in many modified gravity models. To accurately take this into account, one must solve the non-linear partial differential equation (PDE) for the extra degree of freedom of the theory (e.g. the scalar field, $f_{\rm R}$, for the case of $f(R)$ gravity). The `COLASolver` includes the possibility of doing exactly this, but it is quite time-consuming. Instead, we therefore rely on the method of Winther & Ferreira (2015). Here, the Poisson equation is taken to be

$$\Phi(k,z) = \Phi_{\rm N}(k,z) - \frac{3}{2k^2}\Omega_{\rm m} a\delta_{\rm m}^{\rm eff}(k,z)\left(\frac{G_{\rm eff}(k,z)}{G} - 1\right), \tag{16}$$

where the first term is standard Newtonian gravity and the second term is the contribution from the fifth force. The effective density, $\delta_{\rm m}^{\rm eff}$, (in real space) is given by

$$\delta_{\rm m}^{\rm eff}(x,z) = \delta_{\rm m}(x,z)F(\Phi_{\rm N}, \nabla\Phi_{\rm N}, \nabla^2\Phi_{\rm N}, \ldots), \tag{17}$$

where the function F estimates the screening. In this way, $F = 1$ corresponds to no screening. For different models, we can use spherical symmetry to compute the F function. For example, for $f(R)$, we have

$$F = \min\left[1, \frac{3|f_{\rm R0}|}{2|\Phi_{\rm N}|}\left(\frac{\Omega_{\rm m} + 4\Omega_\Lambda}{\Omega_{\rm m} a^{-3} + 4\Omega_\Lambda}\right)^{n+1}\right], \tag{18}$$

which only depends on the local value of the standard Newtonian potential. This is easily (and cheaply) computed in the code using Fourier transforms, making the cost an order of magnitude lower than solving the full equation of motion.

In the `COLASolver`, different screening methods are already implemented for a wide range of models. The above approximation is accurate, but it is not perfect (depending on the model). Because of this, one should always compare the results to full N-body simulations, to assess the accuracy. If higher accuracy is needed, there is a possibility of improving it. One simple fix is to modify the screening method by introducing a fudge factor (or function), $\gamma(a)$, to scale F with. Then, $\gamma(a)$ can be adjusted by comparing to exact simulations. This is done for $f(R)$ in Winther & Ferreira (2015), by fitting a constant factor to match

[2] `https://github.com/HAWinther/FML/tree/master/FML/COLASolver`

[3] `https://github.com/HAWinther/MG-PICOLA-PUBLIC`

a particular redshift. As the main purpose of this paper is to set up a general pipeline, and because emulators for the particular example model used here already exist (e.g. Ramachandra et al. 2021; Arnold et al. 2022; Sáez-Casares et al. 2024), we choose to not adjust $\gamma(a)$ and our screened simulations thus have $\gamma = 1$. The implication this has for the simulations we run is that it overestimates the screening, giving a conservative estimate for the actual boost with respect to ΛCDM.

3.1.3. Massive neutrinos

Massive neutrinos were for a long time considered beyond ΛCDM, at least from the perspective of N-body simulations. This has changed over the last decade, and most simulations these days do include the effect of massive neutrinos.

In the `COLASolver`, massive neutrinos are treated as a field evolving according to linear theory, as proposed by Brandbyge & Hannestad (2009). After creating the CDM+baryon particles, we compute and store the initial density field, $\delta_{\rm cb}(k, z_{\rm ini})$, and evaluate

$$\delta_\nu(k,z) = \frac{T_\nu(k,z)}{T_\nu(k,z_{\rm ini})}\frac{T_\nu(k,z_{\rm init})}{T_{\rm cb}(k,z_{\rm init})}\delta_{\rm cb}(k,z_{\rm ini}), \tag{19}$$

where $T_{\rm cb}$ and T_ν are the CDM+baryon and neutrino transfer functions respectively. This is then added as a source to the Poisson equation (here for GR)

$$\Phi = -\frac{3}{2k^2}\Omega_{\rm m} a\left[(1-f_\nu)\delta_{\rm cdm} + f_\nu\delta_\nu\right], \tag{20}$$

where $f_\nu = \Omega_\nu/\Omega_{\rm m}$. For more information about the neutrino implementation, see Wright et al. (2017). The implementation of massive neutrinos used here is also included in a massive neutrino code comparison project (Euclid Consortium 2023), where it shows percent level agreement in the power spectrum compared to more exact methods of including massive neutrinos.

3.2. Machine learning

To create an emulator for the power spectrum boost, $B(k, z)$, we utilise `PyTorch-Lightening`[4], a lightweight wrapper for the Python `PyTorch` package[5]. `PyTorch` is a machine-learning framework focusing on deep learning, and it provides the tools necessary to train neural networks with multiple layers. It requires our data as input, separated into three different categories: training, testing, and validation. The training data is used to train the neural network. This is the data that the neural network learns from. During the learning process, the neural network occasionally sees the validation data, as a means to help tune the model, but does not learn from it. Once the neural network is fully trained and the emulator is created, it can be evaluated against the test data to assess its performance. The architecture of the neural network training can be designed by the user by deciding the number of hidden dimensions, number of neurons, the batch size, and more[6]. The number of hidden dimensions governs how many layers there are between the input and output layers. Each of these layers has a given number of neurons, which perform computations on the training data before passing it to the next layer. The data is also commonly divided into smaller subsets, containing a set number of samples in each batch. This allows for a more efficient training process.

For our neural network training, we assigned data to the training, validation, and test sets by drawing Latin hypercube samples (McKay et al. 1979; Heitmann et al. 2006) for each data set, corresponding to a 80–10–10 percent distribution. This ensured that each set had an even distribution of parameters in the available parameter set. We also tested both two and three hidden layers, in addition to varying the number of neurons in each layer, ranging from 8–512 in different combinations. Finally, we tested the batch size, varying from 16–256. From our example case of $f(R)$-modified gravity, we created three different emulators, two including unscreened $f(R)$ gravity, one for the linear power spectrum boost and one for the non-linear, and one for the non-linear screened $f(R)$ gravity power spectrum boost. They all had a batch size of 64 and two hidden layers, but the two unscreened $f(R)$ emulators used 16 neurons in the first hidden layer and 8 in the second, while the screened $f(R)$ emulator had 32 and 16 neurons due to a more complicated shape for some of the curves. This can be further optimised and changed by the user based on the desired accuracy of the training process and will depend on the simulations and the parameters that are varied.

3.3. Pipeline

The full pipeline used for this work is made available to the public[7], including instructions on how to use it. It can be applied as is for cosmologies with $f(R)$ modified gravity and massive neutrinos, or extended to different cosmologies as wished. Here follows an outline of the steps necessary both to use the pipeline, and taken inside the pipeline itself:

First of all, the desired cosmological model, if not already included in the `COLASolver`, is implemented. Likewise, the model is implemented in a Boltzmann solver in order to obtain the initial conditions, or an already existing solver with the necessary cosmology can be used. Once this is done, the simulation setup is tested for the model in question to obtain the number of time steps, box size, grid resolution, and so on, that gives the desired convergence within the code itself. With the optimal setup obtained, the boosts dependence on cosmological parameters is tested in order to determine which parameters should be included when creating the emulator. When this is decided, the priors on the parameters is chosen, along with a fiducial cosmology and the number of desired samples to simulate for the neural network to work on. These are the steps that need to be taken outside of the pipeline. Once this is in order, Latin hypercube sampling is employed to sample the parameter space evenly. The way the pipeline is set up now, individual parameter samples are drawn for the training, testing, and validation sets so that they make up an 80–10–10 percentile distribution of the total amount of samples. Alternatively, one can draw all the samples at once and then distribute the samples into data sets later. The desired amount of samples for the various cosmological parameters is written to file, together with the desired simulation settings of the `COLASolver`. The information in this file is then used to generate a bash script where new parameter files for the `COLASolver` are created. This script is then activated and the simulations are run for both the beyond-ΛCDM and ΛCDM model for all the samples. This creates multiple outputs of the matter power spectrum at various redshifts in each case. The boost, $B(k, z) = P_{\text{beyond}-\Lambda\text{CDM}}(k, z)/P_{\Lambda\text{CDM}}(k, z)$,

[4] https://www.pytorchlightning.ai/index.html

[5] https://github.com/pytorch/pytorch, https://pytorch.org/

[6] See the `PyTorch-Lightening` documentation for detailed instructions.

[7] https://github.com/renmau/Sesame_pipeline

is then calculated for each parameter combination and redshift, and smoothed with a Savitzky–Golay filter (Savitzky & Golay 1964). The smoothing is performed to reduce small fluctuations and thereby make the curves easier to estimate for the neural network. The parameter values, redshifts, $\log_{10} k$, and $B(k, z)$ are written into three separate files that go into the neural network learning. 80% of the sample data goes into a training file, 10% into a validation file, and the remaining 10% into a test file, as mentioned above. These are then fed to the neural network, and the power spectrum ratio emulator is created. In this step, the architecture of the neural network must also be decided. This might take some trial and error, in order to obtain the desired accuracy.

At this point, the boost emulator for the desired cosmology has been created. To extract $P_{\mathrm{beyond}-\Lambda\mathrm{CDM}}(k, z)$, we can now depend on already existing high-quality emulators for ΛCDM (e.g. Angulo et al. 2021; Euclid Collaboration 2021; Moran et al. 2023). Ideally, the final step should be to run a high resolution N-body simulation to determine the accuracy of the boost within the `COLASolver`. This would likely be the most expensive part to perform out of everything detailed above, but would give us an estimate of the simulation errors. Alternatively, if high-resolution simulations already exist for the cosmological model of interest, these can be used instead. In addition to this, we can get an estimate of the emulator errors by comparing the emulator performance to the test set. Both of these error estimates can then be baked into the covariance when using the emulator to fit to data, to ensure that all errors are included.

4. Simulations

For the example case of the $f(R)$ model, we performed two main sets of simulations with the `COLASolver` (Sect. 3.1) to obtain the power spectrum boost. One setup had unscreened $f(R)$ gravity, while the other included screening mechanisms. For these runs, we picked 550 samples of five cosmological parameters, varied within the intervals

$$\begin{aligned}
&\sigma_8 \in [0.66, 0.98],\\
&\Omega_{\mathrm{cdm}} \in [0.20, 0.34],\\
&w_0 \in [-1.3, -0.7],\\
&w_{\mathrm{a}} \in [-0.7, 0.7],\\
&\log_{10} f_{\mathrm{R0}} \in [-8.0, -4.0].
\end{aligned} \tag{21}$$

These intervals, with the exception of $\log_{10} f_{\mathrm{R0}}$, are based on the `EuclidEmulator2`, and are either the same (w_0, w_{a}) or slightly larger (Ω_{cdm}, σ_8) than the intervals used by Euclid Collaboration (2021)[8]. A sample selection of 100 parameter samples can be seen in Fig. 1.

The simulation setup and the fiducial cosmology are given in Table 1. In each case, we had $L_{\mathrm{box}} = 350\,h^{-1}$ Mpc, $N_{\mathrm{grid}} = 768$, and $N_{\mathrm{part}} = 640$. The simulations were started at $z_{\mathrm{ini}} = 30.0$ and used 30 timesteps up until $z = 0.0$. Regarding the simulation setup we used, note that COLA simulations in general often use a large force-grid with $N_{\mathrm{grid}} = (2-3)N_{\mathrm{part}}$ (see e.g. Izard et al. 2016). This is in order to have enough force-resolution to be able to create and resolve small halos – a crucial property if one is to create mock galaxy catalogues. The dark matter power-spectrum, on the other hand, is less sensitive to this, and we can therefore get away with using a smaller grid. When it comes to

[8] A convenience of using the `COLASolver` to perform the simulations is that it is fast, and therefore, extending the parameter intervals is computationally cheap compared to more accurate full N-body simulations.

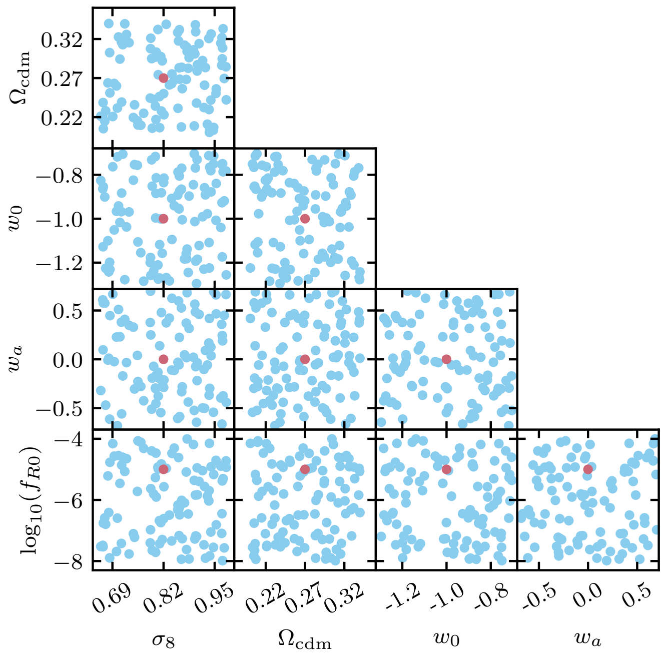

Fig. 1. Sample distribution for 100 of the total 550 samples. The burgundy dot shows the fiducial cosmology parameter values, as given in Table 1.

Table 1. Fiducial values for the main simulations and for parameter variation tests run with a slightly different setup.

Parameter	Fiducial value	Test value
A_{s}	2.1×10^{-9}	2.1×10^{-9}
σ_8	0.82	0.83
n_{s}	0.96	0.96
h	0.67	0.67
M_ν	0.058	0.0
Ω_{cdm}	0.27	0.27
Ω_{b}	0.049	0.05
w_0	−1.0	−1.0
w_{a}	0.0	0.0
$\log_{10} f_{\mathrm{R0}}$	−5.0	−5.0

Notes. If a parameter does not vary, this is its default value. The M_ν parameter refers to the sum of the neutrino masses and is given in eV. The main simulations have $L_{\mathrm{box}} = 350\,h^{-1}$ Mpc, $N_{\mathrm{grid}} = 768$, and $N_{\mathrm{part}} = 640$. The parameter test runs have $L_{\mathrm{box}} = 350\,h^{-1}$ Mpc and $N_{\mathrm{grid}} = N_{\mathrm{part}} = 640$.

choosing the final simulation setup, it is important to always perform convergence tests of how the boost, B, changes with respect to the box size, the number of particles, the force resolution (the grid size), the number of time-steps, and other accuracy parameters like the initial redshift. This is essential to ensure that the result within COLA is converged. This is done for the setup used here, as seen in Fig. 2. Once this is done, the true accuracy can be assessed by comparing the COLA result to high-resolution N-body simulations. The power spectrum boost for the 100 parameter samples mentioned above can be seen in Fig. 3 for three different scenarios; linear boost with unscreened $f(R)$ gravity, non-linear boost with unscreened $f(R)$ gravity, and non-linear boost with screened $f(R)$ gravity.

For every sample, `COLASolver` was run twice, once with $f(R)$-modified gravity and the selected value of f_{R0}, and once

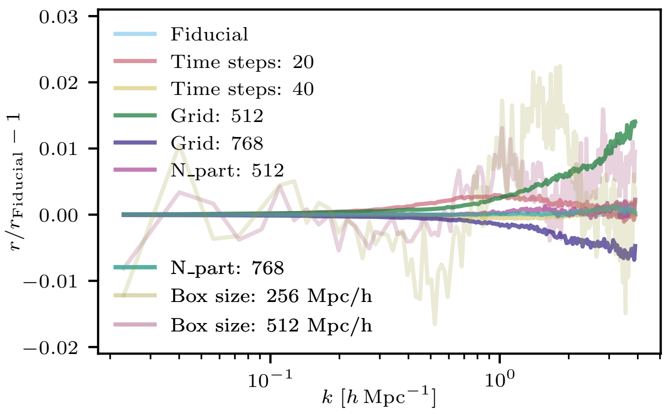

Fig. 2. Convergence test for the `COLASolver` simulation setup for the screened boost between a $f(R)$ gravity simulation with $|f_{R0}| = 10^{-5}$ and $M_\nu = 0.2\,\mathrm{eV}$, and a ΛCDM simulation with massless neutrinos, at $z = 0.0$. This ratio is denoted r, and is shown in comparison to a fiducial simulation setup. The fiducial setup is the same as the test setup in Table 1, namely $N_{\rm time} = 30$, $N_{\rm grid} = N_{\rm part} = 640$, and $L_{\rm box} = 350\,h^{-1}$ Mpc. Here, $N_{\rm time}$ denotes the number of time steps. The only parameter changed from the test setup to the final setup is $N_{\rm grid}$, which was increased to $N_{\rm grid} = 786$ due to the resolution on non-linear scales.

with regular GR. We ran our simulations for GR and $f(R)$ using the same initial conditions (i.e. we used the same value of A_s), which translates into

$$\left(\sigma_8^{\rm f(R)}\right)^2 = \int \frac{k^3}{2\pi^2} P_{\rm GR}(k, z=0) \left(\frac{D_{\rm f(R)}(k,z)}{D_{\rm GR}(z)}\right)^2 \frac{{\rm d}k}{k}, \tag{22}$$

where the growth factors, D, are normalised to unity in the early Universe. This ensures that the boost, B, is unity at early times, while today $\sigma_8^{\rm f(R)}$ is slightly higher for our $f(R)$ simulations than our GR simulations. We used amplitude-fixed initial conditions (Angulo & Pontzen 2016; Villaescusa-Navarro et al. 2018; Klypin et al. 2020) for our simulations to suppress the effects of cosmic variance.

In order to know which cosmological parameter to include as variables in the emulator training, we performed some test simulations. Figure 4 displays the non-linear boost for the unscreened $f(R)$ case, with $|f_{R0}| = 10^{-5}$, when different parameters are allowed to vary. From this, it is clear that σ_8, $\Omega_{\rm cdm}$, w_0, and possibly w_a are the most influential parameters on the power spectrum ratio. Because of this, σ_8, $\Omega_{\rm cdm}$, w_0, w_a, and $\log_{10} f_{R0}$, in addition to z and k, were chosen as the parameters to vary when producing the data used to train the neural network when creating the boost emulator. Tests performed where σ_8 was not kept fixed for the $f(R)$ and GR initial conditions showed a larger variation for all the parameters in general. However, fixing σ_8 showed that some of this effect was due to the difference in clustering. We also performed tests with screened $f(R)$ gravity and a different value for f_{R0}. These tests also pointed toward the same parameter choices.

5. Results

In this section, we present the results for our example boost emulator with $f(R)$-modified gravity for three different cases: the linear power spectrum boost with unscreened $f(R)$ gravity, the non-linear power spectrum boost with unscreened $f(R)$ gravity, and the non-linear power spectrum boost with screened $f(R)$ gravity. The emulator results compared to the test data sets can be seen in Figs. 5–7 respectively, for three different redshifts.

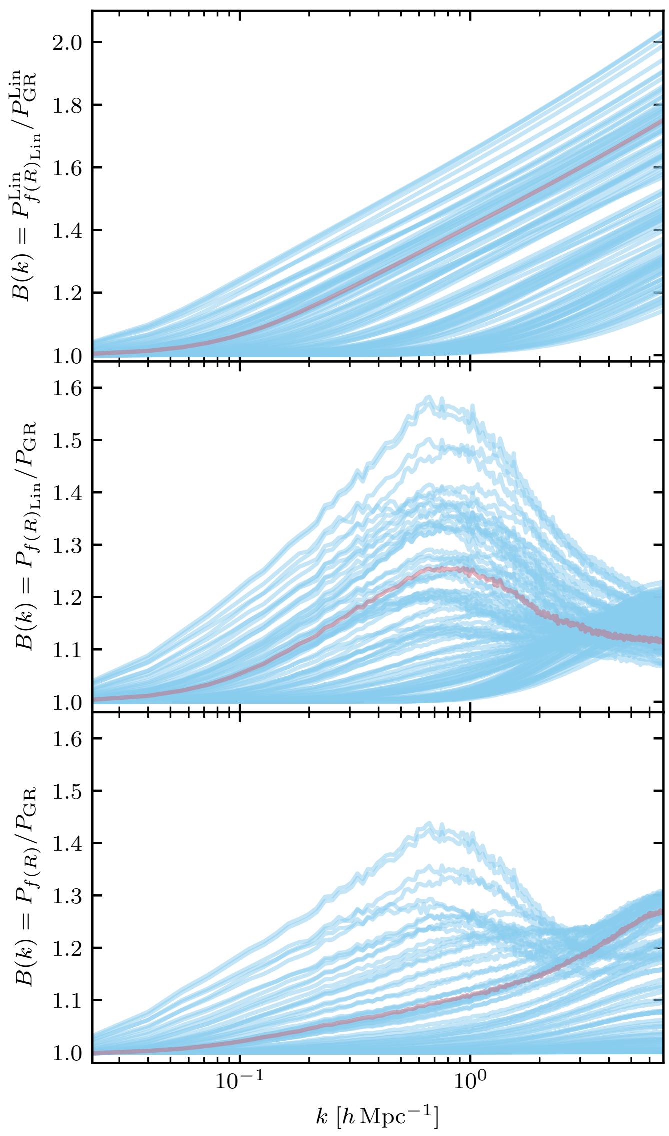

Fig. 3. Matter power spectrum (CDM + baryons) boost for modified gravity and GR for the 100 samples shown in Fig. 1 at $z = 0.0$. The upper and middle panel shows the linear and non-linear boost for unscreened $f(R)$ gravity. The lower panel shows the non-linear boost for screened $f(R)$ gravity. The burgundy line displays the boost for the fiducial cosmology, as listed in Table 1.

In general, we see that the fully linear case has better agreement between the emulator and test data for all redshifts, compared to the non-linear cases. This is most likely due to the simplicity of the boost curve, making it easier for the neural network to predict. The same effect is also seen for higher redshifts in all three cases, where the curves flatten out and become easier for the learning processes to capture accurately. Overall, the fully linear boost emulator agrees with the `COLASolver` simulations to below one percent accuracy on all scales and all redshifts (Fig. 5). For the non-linear unscreened $f(R)$ case (Fig. 6), we have agreement to below or around 1% at all scales for redshifts $z = 1.58$ and $z = 0.55$, with only a few exceptions showing slightly larger discrepancies at small scales. For $z = 0.00$, we have below 1% agreement up to $k \sim 0.3\,h\,{\rm Mpc}^{-1}$, and between 2–2.5% agreement otherwise. For the non-linear boost in the screened $f(R)$ gravity case (Fig. 7), we have below or around one percent accuracy at all scales for $z = 1.58$ and $z = 0.55$. When

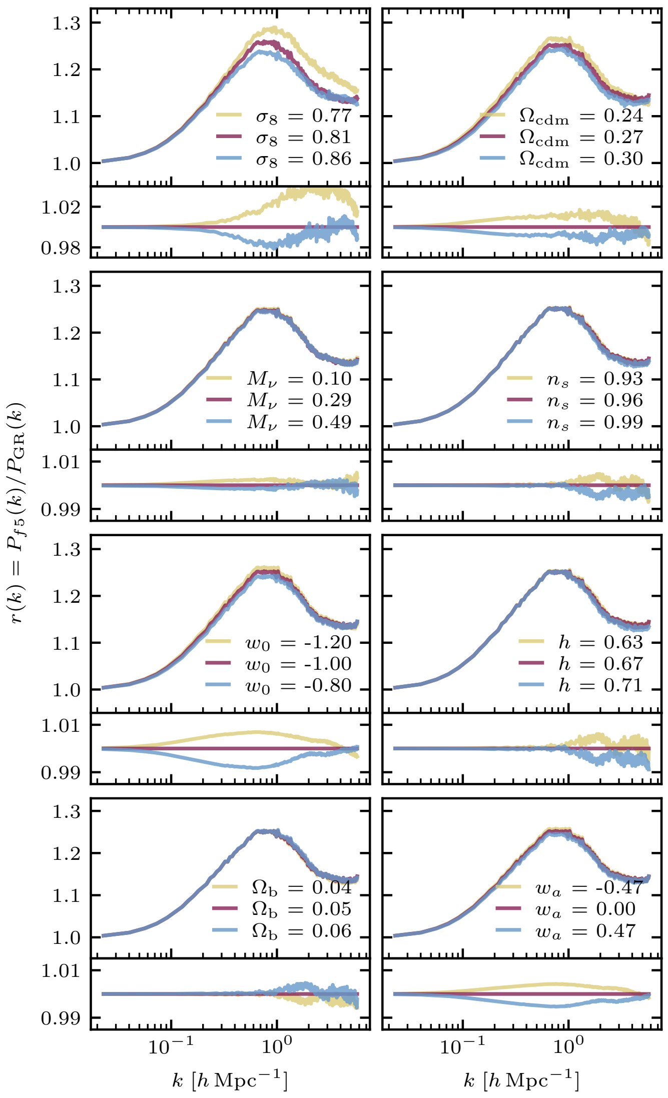

Fig. 4. Boost for modified gravity with $f_{R0} = -10^{-5}$ and GR for different parameter variations. The ratios are shown in the larger panels for three different parameter values, while the narrower panels connected to each large panel show the corresponding ratios of ratios for the three different parameter values, with the middle value as the baseline. The parameters are varied while holding the rest constant, and the fiducial test cosmology is given in Table 1. When varying $\Omega_{\rm cdm}$, $\Omega_{\rm b}$ is kept constant, meaning that $\Omega_{\rm m}$ varies accordingly. When $\Omega_{\rm b}$ is varied, $\Omega_{\rm cdm}$ also varies so that $\Omega_{\rm m}$ is kept constant at a value of 0.32. There is no screening invoked for the $f(R)$ simulations. The M_ν parameter refers to the sum of the neutrino masses and is given in eV. Note the different axes for the first row narrow panels compared to the rest.

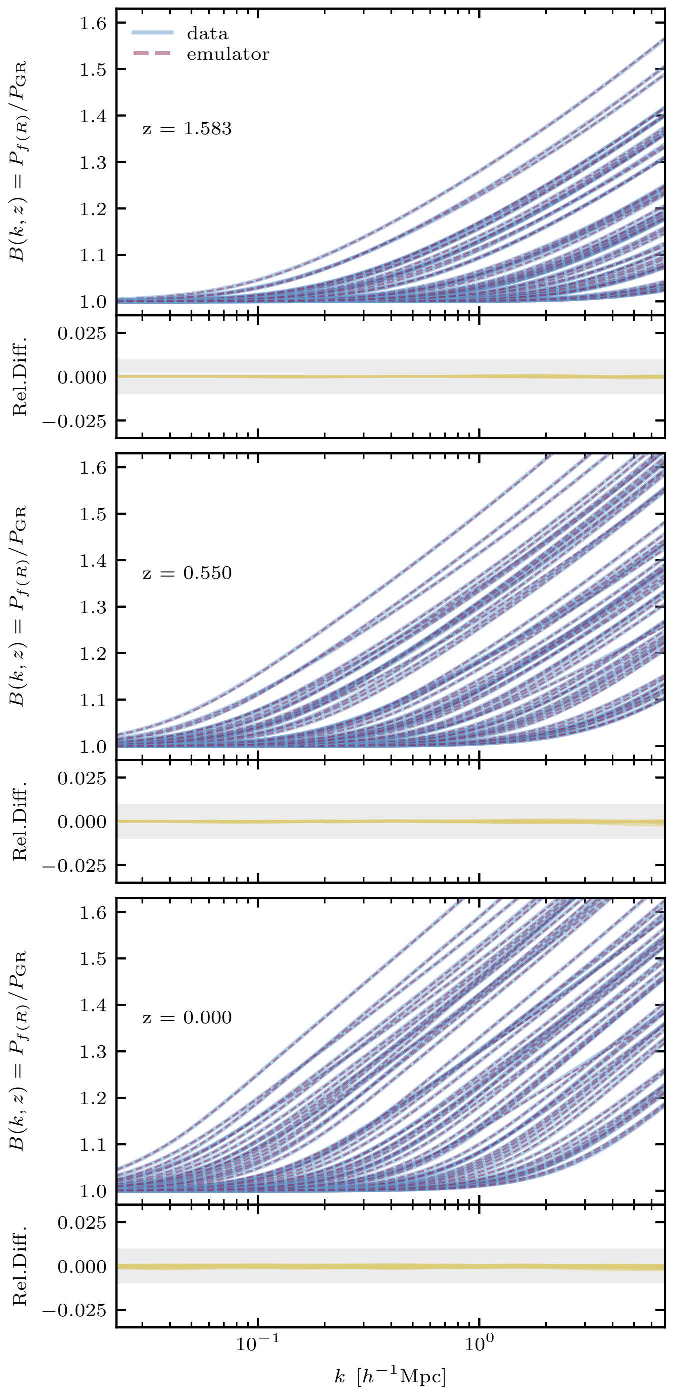

Fig. 5. Emulator performance compared to the test data sets for various redshifts for the linear boost with unscreened $f(R)$ gravity. The emulator results, along with the simulations, are given in the larger panels, while the narrower panels display the corresponding relative difference, given by $B_{\rm emulator}/B_{\rm simulation} - 1$. The grayed-out area in the same panel shows $\pm 1\%$, and the Nyquist frequency of the simulations is $k \approx 5.7\,h\,{\rm Mpc}^{-1}$.

we reach redshift zero there are a few outliers, resulting in some differences around 2.5%, although the bulk of the set stays below 1%. Still, for the non-linear boost emulator, both in the case of screened and unscreened $f(R)$ gravity, it is clear, when compared to Fig. 3, that the curves with the largest discrepancy between predictions and simulations can differ quite a lot from the fiducial expectation. This is not unexpected, as the training set for the neural network contains fewer samples with parameter values that lie close to the edges of the allowed intervals, therefore making the predictions less robust for periphery samples. An example of this is shown in Fig. 8, where an outlier is highlighted. The corresponding parameter sample, compared to the fiducial values, is given in Table 2. The error can be further approved by adjusting the neural network architecture, but this depends on the features in the curve and the parameters included in the training process, and must therefore be adjusted individually for anyone interested in applying the pipeline. It should also be mentioned that for the screened $f(R)$ gravity emulator (Fig. 7), there could be some overfitting for the simplest curves, due to the relatively complex architecture containing 32 and 16 neurons in the two hidden layers. We found that this was necessary in order to catch

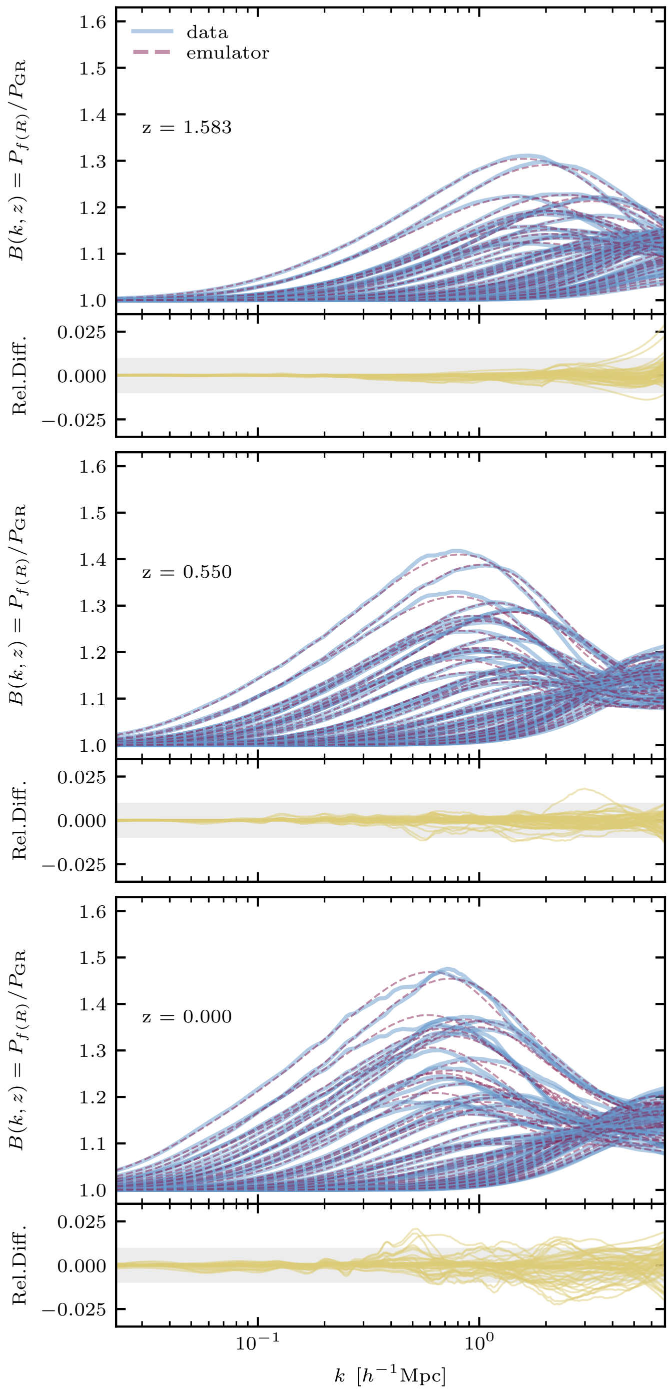

Fig. 6. Emulator performance compared to the test data sets for various redshifts for the non-linear boost with unscreened $f(R)$ gravity. Figure setup as explained for Fig. 5.

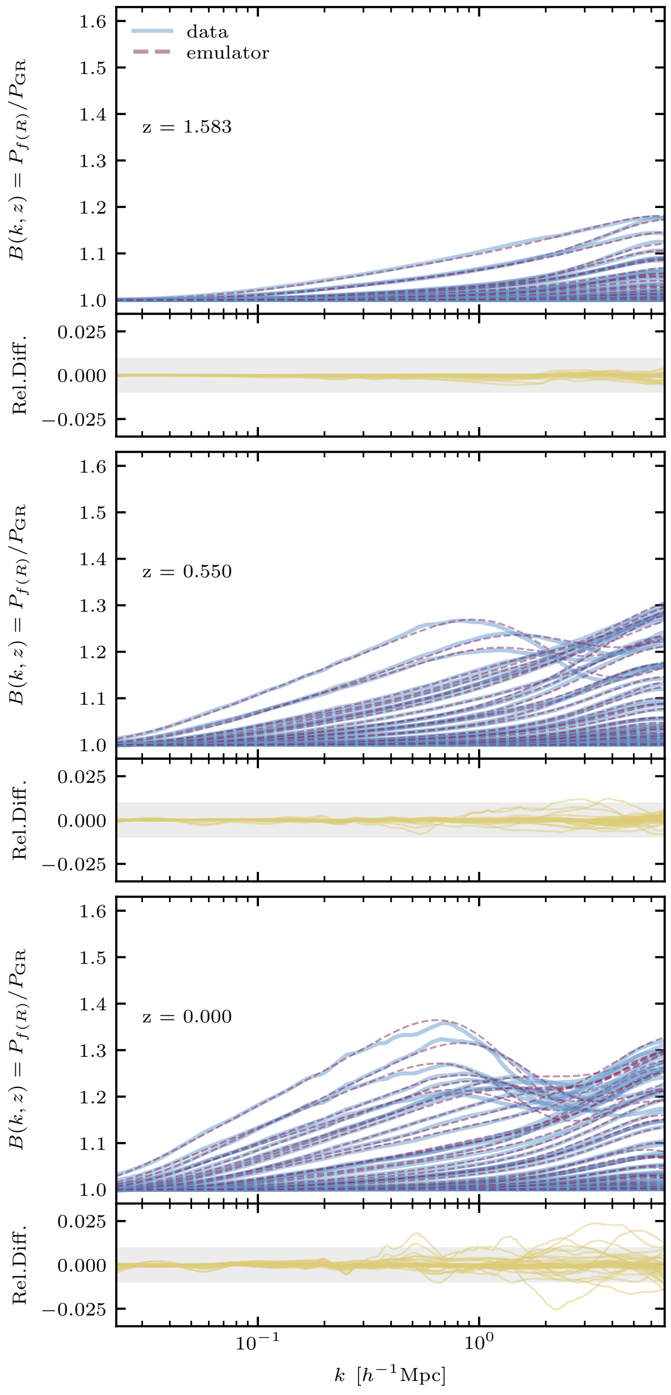

Fig. 7. Emulator performance compared to the test data sets for various redshifts for the non-linear boost with screened $f(R)$ gravity. Figure setup as explained for Fig. 5.

the shape of the more complex curves, like the one highlighted in Fig. 8. This could possibly be remedied by supplying the neural network with smoother data curves.

Finally, in Fig. 9, we make a comparison with a small selection of other emulators in the literature for one set of cosmological parameters (corresponding to the parameters used to make the fitting formula of Winther et al. 2019). Overall, we find a good agreement. The linear emulator is within ∼1% of the linear fitting formula, while the non-linear emulator is within 2−3% of e-Mantis (Sáez-Casares et al. 2024) and the fitting formula. Our no-screening prediction naturally falls between the linear and the non-linear predictions. To our knowledge, there are no other emulators available to compare this to, but for $|f_{\rm R0}| = 10^{-4}$, where screening is not very active, it agrees to a few percent with e-Mantis and the fitting formula. The non-linear emulator we have created here overestimates the screening and gives a conservative estimate for the boost. By properly calibrating the screening efficiency, $\gamma(a)$, in the approximate screening method we use we could improve on this result. As the emulator we created here is mainly an example of the emulator pipeline, we have foregone this step.

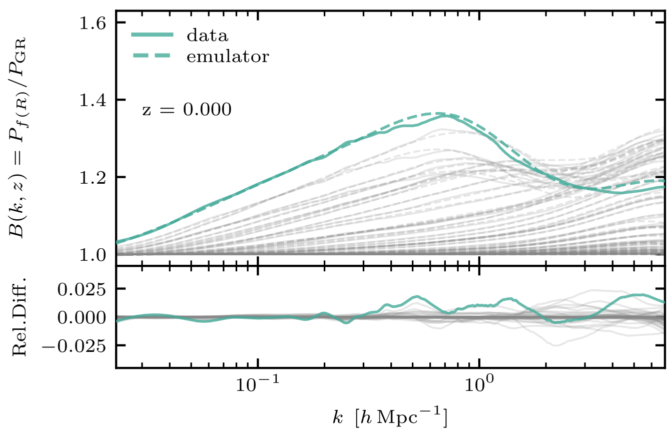

Fig. 8. Lower panel of Fig. 7 with one of the largest outliers highlighted in green. The rest of the results are as before, but muted with a gray colour. The parameter sample values of the outlier can be found in Table 2.

Table 2. Parameters for one of the samples with the largest discrepancy between simulated and emulated boost for the screened $f(R)$ gravity case, compared to the fiducial values.

Parameter	Fiducial sample	Outlier sample
σ_8	0.82	0.87
$\Omega_{\rm cdm}$	0.27	0.21
w_0	−1.0	−1.1
$w_{\rm a}$	0.00	−0.46
$\log_{10} f_{\rm R0}$	−5.00	−4.04

Notes. The parameters $\Omega_{\rm cdm}$, $w_{\rm a}$, and $\log_{10} f_{\rm R0}$ all have values close to the interval boundaries of the emulator training data.

6. Conclusions

Emulators for various global clustering statistics are memory and time-saving. However, creating them often requires a lot of resources through the use of large N-body simulation suites. Because of this, the construction of accurate emulators usually depends on the use of supercomputers. In this paper, we present a full pipeline, Sesame, for creating emulators for the matter power spectrum boost, $B(k,z) = P_{\text{beyond-}\Lambda\text{CDM}}(k,z)/P_{\Lambda\text{CDM}}(k,z)$, for beyond-ΛCDM models, without the need for large computing resources. The pipeline employs the fast and approximate COLA method (Tassev et al. 2013; Wright et al. 2017; Winther et al. 2017) to perform the simulations, simulating both the beyond-ΛCDM and ΛCDM model. This allows us to extract the boost up to higher k-values, due to some of the internal code artifacts canceling, as demonstrated in for example Euclid Consortium (2023). The simulation data is then used to train a neural network, through the `PyTorch Lightening` deep learning module, resulting in a boost emulator. At this point, we rely on existing ΛCDM emulators to extract $P_{\text{beyond-}\Lambda\text{CDM}}(k,z)$.

Using the pipeline will consist of the following steps:

– Implement the model or parametrisation you want to emulate in the `COLASolver`. This most commonly consists of implementing the Hubble function and how to compute the gravitational potential. Here, already implemented models can be used as examples. For most models, this will be a minor task.
– Pick the simulation setup and do a convergence test to ensure that the setup is converged within the code itself.
– Pick which cosmological parameters you are interested in varying. For this, it is useful to study how the boost, B, changes when varying individual cosmological parameters, and select the ones that have a significant impact. From our experience, looking at different modified gravity models that deviate from ΛCDM only close to today, as long as the power (σ_8) is kept the same (depending on the model), either in the initial conditions or at $z = 0$, it is often σ_8 (or $A_{\rm s}$) and $\Omega_{\rm m}$ that are the most relevant.
– Pick the priors of the parameters you want to vary and the number of samples you want to include, and use this to generate the Latin hypercube samples (script provided in the pipeline).
– Generate all the input for `COLASolver`, meaning the input files and the necessary power spectra, by running a Boltzmann solver (script provided in the pipeline for `CLASS` Lesgourgues 2011; Blas et al. 2011).

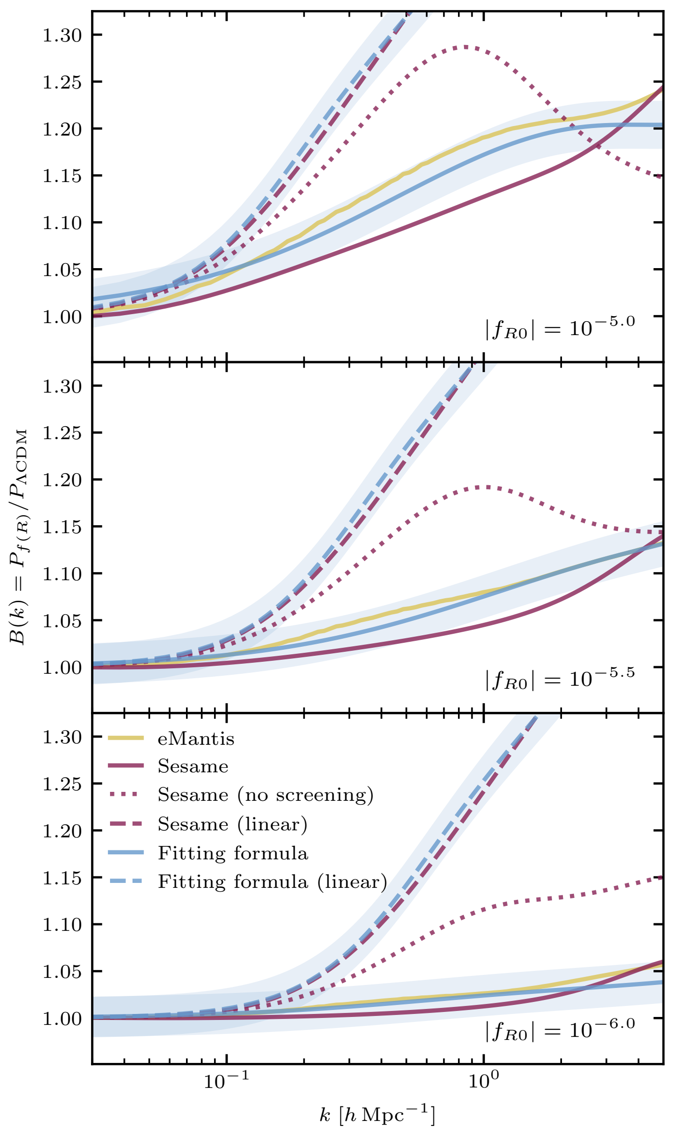

Fig. 9. Comparison of our emulators versus emulators in the literature for $z = 0$. The green line shows e-Mantis (Sáez-Casares et al. 2024) – an emulator based on high-resolution simulations. The red lines show the fitting formula of Winther et al. (2019), which is based on high-resolution simulations for a fixed cosmology. The shaded bands highlight ±2% about the fitting formula prediction.

- Run the simulations to produce all the data files containing the power spectra needed to compute the boosts (script provided in the pipeline).
- Gather all the data and make the files needed for the emulator (script provided in the pipeline).
- Determine the neural network architecture (often trial and error) and run the training to produce the emulator (script provided in the pipeline).
- Check the accuracy of the emulator and redo the previous step if needed until you have something acceptable (script to compare the emulator with data provided in the pipeline).
- Estimate the errors. The emulation error can be obtained from the training set and the error of the simulations themselves can be estimated by running a set of high-resolution N-body simulations or by using already existing simulations.

As an example of using this pipeline, we created three emulators for $f(R)$-modified gravity, including massive neutrinos. The three emulators estimate the boost in the cases of linear and non-linear boost for unscreened $f(R)$ gravity, and the non-linear boost for screened $f(R)$ gravity. The first two of these have not been made before, while for the last case there already exists several emulators (e.g. Ramachandra et al. 2021; Arnold et al. 2022; Sáez-Casares et al. 2024).

The fully linear emulator has below-percent accuracy compared to ground truth, while the non-linear boost emulators have around 1−2% accuracy at redshift zero compared to our simulations and 3−4% accuracy compared to high-resolution simulations. We stress that the emulator with screening that we created here is mainly an example of using the pipeline and is not meant to rival high-quality emulators such as e-Mantis, which is based on high-resolution simulations solving the full $f(R)$ field equation. If this was to be the case, the screening efficiency in the approximate screening method we use for $f(R)$ would have had to be calibrated by comparing to full simulations to enhance the accuracy of the COLA approach. As this is not done here, the non-linear emulator we have created overestimates the screening and gives a lower estimate for the boost. When using emulators to fit data, both the error between the emulator and COLA simulations and that of the COLA method compared to N-body simulations must be taken into account.

With the paper, we provide the full pipeline[9]. Sesame can then be used by anyone to create emulators for their desired beyond-ΛCDM model, either by employing one of the models already incorporated in `COLASolver` code, or by implementing the desired model and then applying the pipeline.

Acknowledgements. We would like to thank the Research Council of Norway for their support.

References

Adamek, J., Durrer, R., & Kunz, M. 2017, JCAP, 2017, 004
Amendola, L. 2000, Phys. Rev. D, 62, 043511
Angulo, R. E., & Pontzen, A. 2016, MNRAS, 462, L1
Angulo, R. E., Zennaro, M., Contreras, S., et al. 2021, MNRAS, 507, 5869
Arnold, C., Li, B., Giblin, B., Harnois-Déraps, J., & Cai, Y. C. 2022, MNRAS, 515, 4161
Baldi, M., Villaescusa-Navarro, F., Viel, M., et al. 2014, MNRAS, 440, 75
Blas, D., Lesgourgues, J., & Tram, T. 2011, JCAP, 2011, 034
Bose, B., Cataneo, M., Tröster, T., et al. 2020, MNRAS, 498, 4650
Bose, B., Wright, B. S., Cataneo, M., et al. 2021, MNRAS, 508, 2479
Bose, B., Tsedrik, M., Kennedy, J., et al. 2023, MNRAS, 519, 4780
Brandbyge, J., & Hannestad, S. 2009, JCAP, 2009, 002
Brando, G., Fiorini, B., Koyama, K., & Winther, H. A. 2022, JCAP, 2022, 051
Brando, G., Koyama, K., & Winther, H. A. 2023, JCAP, 2023, 045
Brax, P., Van De Bruck, C., Davis, A. C., & Shaw, D. J. 2008, Phys. Rev. D, 78, 104021
Buchdahl, H. A. 1970, MNRAS, 150, 1
Bull, P., Akrami, Y., Adamek, J., et al. 2016, Phys. Dark Univ., 12, 56
Cataneo, M., Rapetti, D., Schmidt, F., et al. 2015, Phys. Rev. D, 92, 044009
Chabanier, S., Millea, M., & Palanque-Delabrouille, N. 2019, MNRAS, 489, 2247
Chevallier, M., & Polarski, D. 2001, Int. J. Mod. Phys. D, 10, 213
Clifton, T., Ferreira, P. G., Padilla, A., & Skordis, C. 2012, Phys. Rep., 513, 1
Dakin, J., Brandbyge, J., Hannestad, S., Haugbølle, T., & Tram, T. 2019, JCAP, 2019, 052
de Felice, A., & Tsujikawa, S. 2010, Liv. Rev. Relat., 13, 1
DES Collaboration (Abbott, T. M. C., et al.) 2021, ApJS, 255, 20
DESI Collaboration (Aghamousa, A., et al.) 2016, ArXiv e-prints [arXiv:1611.00036]
Di Valentino, E., Gariazzo, S., & Mena, O. 2021, Phys. Rev. D, 104, 083504
Dodelson, S., & Schmidt, F. 2020, Modern Cosmology (London: Academic Press)
Dvali, G., Gabadadze, G., & Porrati, M. 2000, Phys. Lett. B, 485, 208
Euclid Collaboration (Knabenhans, M., et al.) 2021, MNRAS, 505, 2840
Euclid Collaboration (Scaramella, R., et al.) 2022, A&A, 662, A112
Euclid Consortium (Adamek, J., et al.) 2023, JCAP, 2023, 035
Feng, Y., Chu, M. Y., Seljak, U., & McDonald, P. 2016, MNRAS, 463, 2273
Fiorini, B., Koyama, K., & Izard, A. 2022, JCAP, 2022, 028
Giblin, B., Cataneo, M., Moews, B., & Heymans, C. 2019, MNRAS, 490, 4826
Giocoli, C., Baldi, M., & Moscardini, L. 2018, MNRAS, 481, 2813
Gupta, S., Hellwing, W. A., & Bilicki, M. 2023, Phys. Rev. D, 107, 083525
Hannestad, S., Wong, Y. Y. Y., Thibault, V., et al. 2020, JCAP, 2020, 028
Hassani, F., & Lombriser, L. 2020, MNRAS, 497, 1885
Heitmann, K., Higdon, D., Nakhleh, C., & Habib, S. 2006, ApJ, 646, L1
Heitmann, K., Lawrence, E., Kwan, J., Habib, S., & Higdon, D. 2013, ApJ, 780, 111
Hinterbichler, K., Khoury, J., Levy, A., & Matas, A. 2011, Phys. Rev. D, 84, 103521
Hu, W., & Sawicki, I. 2007, Phys. Rev. D, 76, 064004
Izard, A., Crocce, M., & Fosalba, P. 2016, MNRAS, 459, 2327
J-PAS Collaboration (Benitez, N., et al.) 2014, ArXiv e-prints [arXiv:1403.5237]
Joudaki, S., Ferreira, P. G., Lima, N. A., & Winther, H. A. 2022, Phys. Rev. D, 105, 043522
KATRIN Collaboration (Aker, M., et al.) 2022, Nat. Phys., 18, 160
Khoury, J., & Weltman, A. 2004a, Phys. Rev. Lett., 93, 171104
Khoury, J., & Weltman, A. 2004b, Phys. Rev. D, 69, 044026
Klypin, A., Prada, F., & Byun, J. 2020, MNRAS, 496, 3862
Koda, J., Blake, C., Beutler, F., Kazin, E., & Marin, F. 2016, MNRAS, 459, 2118
Koyama, K. 2016, Rep. Progr. Phys., 79, 046902
Koyama, K., Taruya, A., & Hiramatsu, T. 2009, Phys. Rev. D, 79, 123512
Kwan, J., Heitmann, K., Habib, S., et al. 2015, ApJ, 810, 35
Laureijs, R., Amiaux, J., Arduini, S., et al. 2011, ArXiv e-prints [arXiv:1110.3193]
Leclercq, F., Faure, B., Lavaux, G., et al. 2020, A&A, 639, A91
Lesgourgues, J. 2011, ArXiv e-prints [arXiv:1104.2932]
Lesgourgues, J., & Pastor, S. 2006, Phys. Rep., 429, 307
Li, B., Zhao, G. B., Teyssier, R., & Koyama, K. 2012, JCAP, 2012, 051
Linder, E. V. 2003, Phys. Rev. Lett., 90, 4
Liu, J., Bird, S., Matilla, J. M. Z., et al. 2018, JCAP, 2018, 049
Llinares, C., Mota, D. F., & Winther, H. A. 2014, A&A, 562, A78
LSST Collaboration (Ivezic, Z., et al.) 2019, ApJ, 873, 111
Marsh, D. J. 2016, Phys. Rep., 643, 1
Martin, J., Ringeval, C., & Vennin, V. 2014, Phys. Dark Univ., 5, 75
Mauland, R., Elgarøy, Ø., Mota, D. F., & Winther, H. A. 2023, A&A, 674, A185
McKay, M. D., Beckman, R. J., & Conover, W. J. 1979, Technometrics, 21, 239
Moran, K. R., Heitmann, K., Lawrence, E., et al. 2023, MNRAS, 520, 3443
Nishimichi, T., Takada, M., Takahashi, R., et al. 2019, ApJ, 884, 29
Particle Data Group (Workman, R. L., et al.) 2022, Progr. Theor. Exp. Phys., 2022, 083C01
Partmann, C., Fidler, C., Rampf, C., & Hahn, O. 2020, JCAP, 2020, 018
Peebles, E. P. J. 1980, The Large-Scale Structure of the Universe (Princeton: Princeton University Press)
Planck Collaboration VI. 2020, A&A, 641, A6
Pogosian, L., & Silvestri, A. 2008, Phys. Rev. D, 77, 023503
Potter, D., Stadel, J., & Teyssier, R. 2017, Comput. Astrophys. Cosmol., 4, 2
Puchwein, E., Baldi, M., & Springel, V. 2013, MNRAS, 436, 348
Ramachandra, N., Valogiannis, G., Ishak, M., & Heitmann, K. 2021, Phys. Rev. D, 103, 123525
Ruan, C. Z., Hernández-Aguayo, C., Li, B., et al. 2022, JCAP, 2022, 018
Sáez-Casares, I., Rasera, Y., & Li, B. 2024, MNRAS, 527, 7242

[9] https://github.com/renmau/Sesame_pipeline

Savitzky, A., & Golay, M. 1964, J. Anal. Chem., 36, 1627
Song, Y. S., Hu, W., & Sawicki, I. 2007, Phys. Rev. D, 75, 044004
Sotiriou, T. P., & Faraoni, V. 2010, Rev. Mod. Phys., 82, 451
Springel, V., Pakmor, R., Zier, O., & Reinecke, M. 2021, MNRAS, 506, 2871
Tassev, S., Zaldarriaga, M., & Eisenstein, D. J. 2013, JCAP, 2013, 036
Tassev, S., Eisenstein, D. J., Wandelt, B. D., & Zaldarriaga, M. 2015, ArXiv e-prints [arXiv:1502.07751]
Thomson, M. 2013, Modern Particle Physics (Cambridge: Cambridge University Press)
Valogiannis, G., & Bean, R. 2017, Phys. Rev. D, 95, 103515
Villaescusa-Navarro, F., Naess, S., Genel, S., et al. 2018, ApJ, 867, 137
Weinberger, R., Springel, V., & Pakmor, R. 2020, ApJS, 248, 32
Wetterich, C. 1988, Nucl. Phys. B, 302, 668
Will, C. M. 2014, Liv. Rev. Relat., 17, 1
Winther, H. A., & Ferreira, P. G. 2015, Phys. Rev. D, 91, 123507
Winther, H. A., Schmidt, F., Barreira, A., et al. 2015, MNRAS, 454, 4208
Winther, H. A., Koyama, K., Manera, M., Wright, B. S., & Zhao, G. B. 2017, JCAP, 2017, 006
Winther, H. A., Casas, S., Baldi, M., et al. 2019, Phys. Rev. D, 100, 123540
Wright, B. S., Winther, H. A., & Koyama, K. 2017, JCAP, 2017, 054
Wright, B. S., Sen Gupta, A., Baker, T., Valogiannis, G., & Fiorini, B. 2023, JCAP, 2023, 040
Zhao, G. B. 2014, ApJS, 211, 23