# ECOSMOG: An Efficient Code for Simulating Modified Gravity

Baojiu Li,[1,2,3,4,*] Gong-Bo Zhao,[5,†] Romain Teyssier,[6,7,‡] and Kazuya Koyama[5,§]

[1] *ICC, Physics Department, University of Durham, South Road, Durham DH1 3LE, UK*
[2] *DAMTP, Centre for Mathematical Sciences, University of Cambridge, Wilberforce Road, Cambridge CB3 0WA, UK*
[3] *Kavli Institute for Cosmology Cambridge, Madingley Road, Cambridge CB3 0HA, UK*
[4] *Institute of Astronomy, Madingley Road, Cambridge CB3 0HA, UK*
[5] *Institute of Cosmology and Gravitation, University of Portsmouth, Portsmouth, PO1 3FX, UK*
[6] *Universität Zürich, Institute für Theoretische Physik, Winterthurerstrasse 190, CH-8057 Zürich, Switzerland*
[7] *CEA Saclay, DSM/IRFU/SAP, Bâtiment 709, F-91191 Gif-sur-Yvette, Cedex, France*

(Dated: October 10, 2011)

We introduce a new code, ECOSMOG, to run $N$-body simulations for a wide class of modified gravity and dynamical dark energy theories. These theories generally have one or more new dynamical degrees of freedom, the dynamics of which are governed by their (usually rather nonlinear) equations of motion. Solving these non-linear equations has been a great challenge in cosmology. Our code is based on the RAMSES code, which solves the Poisson equation on adaptively refined meshes to gain high resolutions in the high-density regions. We have added a solver for the extra degree(s) of freedom and performed numerous tests for the $f(R)$ gravity model as an example to show its reliability. We find that much higher efficiency could be achieved compared with other existing mesh/grid-based codes thanks to two new features of the present code: (1) the efficient parallelisation and (2) the usage of the multigrid relaxation to solve the extra equation(s) on *both* the regular domain grid *and* refinements, giving much faster convergence even under much more stringent convergence criteria. This code is designed for performing high-accuracy, high-resolution and large-volume cosmological simulations for modified gravity and general dark energy theories, which can be utilised to test gravity and the dark energy hypothesis using the upcoming and future deep and high-resolution galaxy surveys.



## I. INTRODUCTION

The accelerating expansion of our Universe is one of the most challenging questions in modern physics [1]. After more than a decade of attempts in constructing consistent and natural theories for it, we are still nowhere near a definite conclusion. Broadly speaking, the theoretical models developed so far roughly fall into two categories: those involving some exotic matter species, the so-called dark energy, which usually have nontrivial dynamics, and those involving modifications to the Einstein gravity on certain (usually large) scales. Examples of the dynamical dark energy include the quintessence [2], k-essence ams2000, coupled quintessence [4], chameleon field [5, 6], symmetron field [7] *etc.*, and some examples of modified gravity models are the $f(R)$ gravity [8], scalar-tensor theory [9] and DGP model [10] (for some recent review see, *e.g.*, [11–13]).

From a practical point of view (*i.e.*, regardless of the naturalness or fine-tuning considerations), there are two important issues faced by any theoretical model: consistency and degeneracy.

In order to ensure the consistency of the mode, a given model should not violate any existing observational constraints. The new degrees of freedom, which are supposed to drive the accelerating expansion only on very large scales, quite often produce unwanted side effects. Consider the coupled quintessence as an example; if it couples to normal matter species (baryons *etc.*), it can mediate a fifth force that is strongly constrained by experiments. This problem can certainly be avoided by assuming that the quintessence field only couples to dark matter which we cannot directly observe, but more interesting theoretical mechanisms to avoid this problem have been designed in recent years. The leading example is the chameleon mechanism [5, 6, 14]. By this mechanism, the fifth force is suppressed to the undetectable level in regions with high matter density, while in low-density environments such as galaxies, galaxy clusters and cosmological voids, the fifth force is unsuppressed and can be as strong as gravity. The rapid changes of the behaviour of the fifth force across different regions naturally imply that the equation(s) of motion governing the dynamics of the new degree(s) of freedom should be very nonlinear. This adds the complexity to the model but we cannot avoid this to ensure the consistency of the model.

Different theoretical models can behave very similarly to each other and to the standard ΛCDM paradigm, which makes it difficult to distinguish amongst them using observational data. The (apparent) existence of the degeneracy usually indicates the incompleteness of our theoretical understanding of models. For instance, it is very easy to design a scalar field model which produces almost identical cosmic expansion history as ΛCDM, but this is merely because in the background cosmology, any detailed structure of the Universe is disregarded; when we

* Email address: b.li@damtp.cam.ac.uk
† Email address: gong-bo.zhao@port.ac.uk
‡ Email address: teyssier@physik.uzh.ch
§ Email address: kazuya.koyama@port.ac.uk

look at the linear perturbation evolutions, the apparent degeneracy is often broken. However, there are situations where the degeneracy cannot be broken even with linear perturbation analysis (such as the chameleon theory and $f(R)$ gravity in which the length scales on which linear perturbation analyses apply are bigger than the range of, and so not affected by, the fifth force). In such situations, a full study of the nonlinear structure formation and evolution needs to be performed to break the degeneracy.

Therefore, it is essential to study non-linea structure formation to ensure the consistency of the model and break the degeneracies between various models. Nonlinear structure formation is too complex to understand analytically and therefore requires numerical simulations. Numerical simulations of the cosmic structure evolution on small (such as galactic or cluster) scales can not only open a new arena of using consistency with observations to constrain models, but also hopefully break the degeneracy between different models (as our cosmological simulations show below). This is particularly exciting considering that high-quality observational data will keep coming in the next decades to improve our theoretical understanding.

Numerical simulations are done by numerical codes. In most cases the extra dynamical degree(s) of freedom is more accurately treated as a field and its value is required on a number of space points. For this purpose we need the numerical code that can solve the field value on meshes covering (parts of) the simulation box. There are two such codes known to us to date: one is that of Oyaizu [15], which has been applied to $f(R)$ gravity [16, 17] and the DGP model [18]; the other is a modified `MLAPM` code [19] written by one of the authors [20] and which has been applied to chameleon theories [21], $f(R)$ gravity [22], coupled scalar field theories [23, 24], scalar-tensor theory [25], dilaton model [26], symmetron model [27]. Both have shortcomings: the Oyaizu's code does not support adaptive refinements (thus easier to implement) and high-resolution simulations are not practical; while the `MLAPM` code does support adaptive refinements, it is not parallelised and not practical for simulations with very big volumes and high mass resolutions. Furthermore, the equations on `MLAPM` refinements are solved on a one-level grid, which is rather inefficient.

The aim of this paper is to introduce a new code `ECOSMOG`, which overcomes the shortcomings of the previous codes. The new code is based on `RAMSES` [28]. It is efficiently parallelised, supports adaptive refinements and solves the equations using multigrid method on the refinements (for a more detailed comparison of the three codes the readers can refer to the table I). As a working example, we use the new code to run a number of test simulations for the $f(R)$ gravity, which is one of the most challenging models to simulate because of the high nonlinearity of its equations. As will be shown below, the code works very well for the $f(R)$ model, and we certainly expect it to be straightforward to implement equations in other models to our code.

The organisation of this paper is as follows. In § II we briefly introduce the $f(R)$ gravity model. In § III we introduce the supercomoving code unit using a different form and list the $N$-body Poisson and $f(R)$ equations to simplify the numerics. § IV then makes discrete versions of these equations to be implemented in our code. § V, we show details on how the numerical implementation is performed and discuss several important issues, which is the core section of this paper. Next, a long section, § VI, contains the results of eleven tests of the code. These tests check the correctness, efficiency and consistency of different aspects of the code and give us confidence about its reliability. Finally we compare the present code with other mesh-based codes, summarise and conclude in § VII.

This is a paper to explain the code and physical interpretations of the results will be presented in future publications.

## II. A TEST CASE: THE $f(R)$ GRAVITY

One can alter the Einstein gravity in such a way that it gives rise the cosmic acceleration without introducing dark energy. One example along this line is the so-called $f(R)$ gravity, in which the Ricci scalar $R$ in the Einstein-Hilbert action is generalised to a function of $R$ (see *e.g.*, [13] for a review and references therein). In $f(R)$ gravity, the structure formation is governed by the following two equations,

$$\nabla^2\Phi = \frac{16\pi G}{3}a^2\delta\rho_{\rm M} + \frac{a^2}{6}\delta R(f_R), \tag{1}$$

$$\nabla^2\delta f_R = -\frac{a^2}{3}[\delta R(f_R) + 8\pi G\delta\rho_{\rm M}], \tag{2}$$

where $\Phi$ denotes the gravitational potential, $f_R \equiv \frac{{\rm d}f(R)}{{\rm d}R}$ is the extra scalar degree of freedom, dubbed *scalaron*, $\delta f_R = f_R(R) - f_R(\bar{R}), \delta R = R - \bar{R}, \delta\rho_{\rm M} = \rho_{\rm M} - \bar{\rho_{\rm M}}$, and the quantities with overbar take the background values. The symbol $\nabla$ is the three dimensional gradient operator, and $a$ is the scale factor. These two coupled Poisson-like equations are much difficult to solve than the single Poisson equation in General Relativity (GR), $\nabla^2\Phi = 4\pi G a^2\delta\rho_{\rm M}$, which is linear. From Eqs (1) and (2), we can see that gravity in $f(R)$ can be enhanced depending on the local environment – in underdense regions, the $\delta R(f_R)$ term in Eqs (1) vanishes thus the two equations decouple, making gravity simply enhanced by a factor of 4/3. However in the dense region, $\delta f_R$ in Eq (2) is negligible, yielding $\delta R(f_R) = -8\pi G\delta\rho_{\rm M}$, which means that GR is locally restored. This is the chameleon mechanism applied in the $f(R)$ gravity models, which is important for the cosmological viability of the latter. The presence of the chameleon effect indicates that Eqs (1) and (2) are highly nonlinear, making it challenging to numerically solve them in the simulation process.

An $f(R)$ gravity model is fully specified by the functional form of $f(R)$, and here we shall adopt the model proposed by Hu & Sawicki [29], which takes the form

$$f(R) = -m^2 \frac{c_1(-R/m^2)^n}{c_2(-R/m^2)^n + 1}, \quad (3)$$

where $n, c_1, c_2$ are model parameters, and

$$m^2 \equiv \Omega_m H_0^2, \quad (4)$$

with $\Omega_m$ being the present fractional matter density and $H_0$ the current Hubble expansion rate.

It can then be shown that

$$f_R = -\frac{c_1}{c_2^2} \frac{n(-R/m^2)^{n+1}}{[(-R/m^2)^n + 1]^2}. \quad (5)$$

Because

$$-\bar{R} \approx 8\pi G\bar{\rho}_m - 2\bar{f}(R) = 3m^2 \left[a^{-3} + \frac{2}{3}\frac{c_1}{c_2}\right], \quad (6)$$

where overbars are used for the background quantities, to match a ΛCDM background expansion we have $c_1/c_2 = 6\Omega_\Lambda/\Omega_m$. With $\Omega_m = 0.24$ and $\Omega_\Lambda = 0.76$, we find $-\bar{R} \approx 41m^2 \gg m^2$, which means that $f_R$ can be approximated as

$$f_R \approx -n\frac{c_1}{c_2^2}\left[\frac{m^2}{-R}\right]^{n+1}. \quad (7)$$

Because $f_R$ is the actual quantity that enters the $N$-body equations (see below), we find that only the two parameters $n$ and $\xi \equiv c_1/c_2^2$, are needed in our simulations. Another (independent) parameter $f_{R0}$ which is the present background value of $f_R$, can be obtained from $\xi$ and is often used to give people some idea about the size of $f_R$.

## III. THE $N$-BODY EQUATIONS

In this section we shall introduce the convention of our code units, and list the $N$-body Poisson and $f(R)$ equations, the latter being written in the so-called quasi-static limit so that terms involving time derivatives will be dropped. The $N$-body equations can all be found elsewhere, though perhaps of slightly different forms; consequently, this section shall be kept short and only serves for completeness.

### A. The Code Units

The code unit used in the `RAMSES` code and its modification developed here is based on (but not exactly) the supercomoving coordinates of [30]. It can be summarised as follows (where tilded quantities are expressed in the code unit):

$$\tilde{x} = \frac{x}{aB}, \quad \tilde{\rho} = \frac{\rho a^3}{\rho_c \Omega_m}, \quad \tilde{v} = \frac{av}{BH_0},$$
$$\tilde{\phi} = \frac{a^2\phi}{(BH_0)^2}, \quad d\tilde{t} = H_0\frac{dt}{a^2}, \quad \tilde{c} = \frac{c}{BH_0}.$$

In the above $x$ is the comoving coordinate, $\rho_c$ is the critical density today, $\Omega_m$ the fractional energy density for matter today, $v$ the particle velocity, $\phi$ the gravitational potential and $c$ the speed of light. In addition, $B$ is the size of the simulation box in unit of $h^{-1}$Mpc and $H_0$ the Hubble expansion rate today in unit of $100h$ km/s/Mpc. Note that with these conventions the average matter density is $\tilde{\bar{\rho}} = 1$. All the newly-defined quantities are dimensionless.

In the $f(R)$ gravity equation we also have a new degree of freedom $f_R$ and in the code unit we will use $\tilde{f}_R \equiv a^2 f_R$ instead. As we shall see below, this is to make sure that $\tilde{f}_R$ has an equivalent status as $\tilde{\phi}$: the latter is the Newtonian potential and determines the total (modified) gravitational force, while the former is related to the potential of the extra force and determines the modification to the standard Einsteinian gravity.

### B. The Modified Poisson Equation

From above we see that the modified Poisson equation is

$$\vec{\nabla}^2\phi = \frac{16\pi G}{3}\delta\rho - \frac{1}{6}\delta R \quad (8)$$

where $\delta R = R - \bar{R}$ and

$$R = m^2\left(-\frac{n\xi}{f_R}\right)^{\frac{1}{n+1}}, \quad (9)$$

$$\bar{R} = 3m^2\left(a^{-3} + 4\frac{\Omega_\Lambda}{\Omega_m}\right). \quad (10)$$

Using the code units defined above, the equation becomes

$$\tilde{\nabla}^2\tilde{\phi} = 2\Omega_m a\left(\tilde{\rho} - 1\right) - \frac{1}{6}\Omega_m a^4\left[\left(-\frac{na^2\xi}{\tilde{f}_R}\right)^{\frac{1}{n+1}} - 3\left(a^{-3} + 4\frac{\Omega_\Lambda}{\Omega_m}\right)\right] \quad (11)$$

### C. The $f(R)$ Equation

The equation of motion for $f_R$,

$$\vec{\nabla}^2 f_R = \frac{1}{3c^2}\left[\delta R - 8\pi G\delta\rho\right], \quad (12)$$

becomes in the code units,

$$\tilde{\nabla}^2\tilde{f}_R = -\frac{1}{3\tilde{c}^2}\Omega_m a\left(\tilde{\rho} - 1\right) + \frac{1}{3\tilde{c}^2}\Omega_m a^4\left[\left(-\frac{na^2\xi}{\tilde{f}_R}\right)^{\frac{1}{n+1}} - 3\left(a^{-3} + 4\frac{\Omega_\Lambda}{\Omega_m}\right)\right] \quad (13)$$

Comparing this equation with that for the modified Poisson equation, we find that $\tilde{\phi}$ and $\tilde{c}^2\tilde{f}_R$ has the same code unit, and that is why we have used $\tilde{f}_R$ instead of $f_R$.

As in [22], we will not solve $\tilde{f}_R$ directly as it may change too quickly from one space point to another and can cause divergence problems in the numerical solutions. Instead, we will solve a new variable $\tilde{u}$, defined through $\tilde{f}_R \equiv -e^{\tilde{u}}$. $\tilde{u}$ will be of order unity across the space, and so has better convergence properties.

## IV. THE DISCRETISED EQUATIONS

From here on we will only use variables expressed using the code unit, and the tildes will be dropped for simplicity.

Clearly, to put the above equations into the $N$-body code one must discretise them. For the Poisson equation, it is linear as along as the source term (the right-hand side) is provided, so the discretisation is fairly straightforward:

$$\begin{aligned}&\frac{1}{h^2}\left[\phi_{i+1,j,k}+\phi_{i-1,j,k}+\phi_{i,j+1,k}+\phi_{i,j-1,k}+\phi_{i,j,k+1}+\phi_{i,j,k-1}-6\phi_{i,j,k}\right]\\&=2\Omega_m a\left(\delta_{i,j,k}-1\right)-\frac{1}{6}\Omega_m a^4\left[\left(na^2\xi\right)^{\frac{1}{n+1}}\exp\left(-\frac{u_{i,j,k}}{n+1}\right)-3\left(a^{-3}+4\frac{\Omega_\Lambda}{\Omega_m}\right)\right],\end{aligned}\tag{14}$$

where for example $\phi_{i,j,k}$ is the value of $\phi$ in the grid cell with index $(i,j,k)$. The discrete version of the nonlinear $f(R)$ equation of motion, in contrast, is more tedious [22]:

$$\begin{aligned}&\frac{1}{h^2}\left[b_{i+\frac{1}{2},j,k}u_{i+1,j,k}-u_{i,j,k}\left(b_{i+\frac{1}{2},j,k}+b_{i-\frac{1}{2},j,k}\right)+b_{i-\frac{1}{2},j,k}u_{i-1,j,k}\right]\\&+\frac{1}{h^2}\left[b_{i,j+\frac{1}{2},k}u_{i,j+1,k}-u_{i,j,k}\left(b_{i,j+\frac{1}{2},k}+b_{i,j-\frac{1}{2},k}\right)+b_{i,j-\frac{1}{2},k}u_{i,j-1,k}\right]\\&+\frac{1}{h^2}\left[b_{i,j,k+\frac{1}{2}}u_{i,j,k+1}-u_{i,j,k}\left(b_{i,j,k+\frac{1}{2}}+b_{i,j,k-\frac{1}{2}}\right)+b_{i,j,k-\frac{1}{2}}u_{i,j,k-1}\right]\\&+\frac{1}{3\tilde{c}^2}\Omega_m a^4\left(na^2\xi\right)^{\frac{1}{n+1}}\exp\left(-\frac{u_{i,j,k}}{n+1}\right)-\frac{1}{\tilde{c}^2}\Omega_m a\left(\delta_{i,j,k}-1\right)-\frac{1}{\tilde{c}^2}\Omega_m a^4\left(a^{-3}+4\frac{\Omega_\Lambda}{\Omega_m}\right)\;=\;0.\end{aligned}\tag{15}$$

Notice that in the above equations we have used $\delta_{i,j,k} \equiv \tilde{\rho}_{i,j,k}$ to avoid confusion in notations and to make it clear that we are using the density *contrast*. $b \equiv e^u$ and

$$\begin{aligned}b_{i+\frac{1}{2},j,k} &\equiv \frac{1}{2}\left(b_{i+1,j,k}+b_{i,j,k}\right),\\ b_{i-\frac{1}{2},j,k} &\equiv \frac{1}{2}\left(b_{i,j,k}+b_{i-1,j,k}\right),\;\cdots\end{aligned}$$

## V. THE $N$-BODY ALGORITHMS

The detailed implementation of the $N$-body solvers in the `RAMSES` code is indeed quite different from that in the `MLAPM` code [20, 23]. As a result, certain corrections need to be made to the above discrete equations before implementing them.

Both codes adaptively refine the grid in high-density regions and solve the Poisson and $f(R)$ equations on the refinement to get better spatial resolutions. The refining is performed on a grid-by-grid basis so that the final refinements typically have irregular shapes roughly matching the iso-density surfaces. Particles in regions underlying the refinements are linked to the latter, where their positions and momenta are updated.

An important difference between the two codes is how the equations are solved on the refinements. `MLAPM` solves the equations on single level of the considered refinement only, performing Gauss-Seidel relaxations till the convergence of the solution is obtained. In `RAMSES`, however, for each adaptive mesh refinement (AMR) level there is a series of separate, coarser multigrid levels on which the Gauss-Seidel relaxation is used to accelerate the convergence.

The treatments of boundary conditions on refinements are also different. In `MLAPM`, the outermost cells of a refinement are taken as its physical boundary, thus the field values wherein are set using interpolation from the coarser-levels solutions. In `RAMSES`, the use of multilevel requires the boundary conditions to be set consistently on the different levels, and this is achieved using an elegant *mask* scheme: on the AMR level, which is the finest level in the multigrid series, the active cells, inside which the field values need to be updated during the relaxation process, are given a mask value of +1 while the other

cells are assigned a mask value $-1$. The boundary of the computational domain is defined to be where the linearly-interpolated mask value vanishes, or equally the boundary of the active AMR cells. The value of the field on this physical boundary is computed at the beginning of the relaxation process and kept unchanged after that until a converged solution in the active cells are obtained. The mask values in the coarser multigrid cells are restricted from their finer-level values, while the physical boundary on this level is still defined as where the mask hits zero, which ensures the boundaries on different levels are consistent (for more details and tests in the classical GR case see [33]).

Moreover, the treatments of the boundary conditions for linear and nonlinear elliptical PDEs are also different in `RAMSES`, as we shall see below.

### A. The Modified Poisson Equation

#### 1. The BC-modified Equation

Unlike the `MLAPM` code, where the refinement boundary consists of the outermost cells on that refinement and so the boundary cells are always there, in `RAMSES` we might well face the situation that an outermost active cell has no neighbouring cells in some directions on the save level (possibly because a neighbouring coarse cell has not been refined), while we do need the field values in those cells to interpolate and compute the boundary conditions. Such cells are called ghost cells: they do not exist in the code data structure but we do need them.

For the linear elliptical PDEs, such as the (modified) Poisson equation, there is a simple way to avoid storing information about the ghost cells. Consider the discrete Poisson equation and now suppose that the $(i-1,j,k)$-cell is a ghost, which has mask value $m_{i-1,j,k}$[1]. We realise that the physical boundary, namely where the mask value crosses zero, will be a distance

$$\frac{-m_{i-1,j,k}}{m_{i,j,k}-m_{i-1,j,k}}h$$

from the $(i-1,j,k)$ (ghost!) cell and

$$\frac{m_{i,j,k}}{m_{i,j,k}-m_{i-1,j,k}}h$$

from the $(i,j,k)$-cell, where $h$ is the cell length, or equally the distance between the centres of these two cells. Using linear interpolation, the boundary value of $\phi$ will then be

$$\phi_b=\frac{m_{i,j,k}}{m_{i,j,k}-m_{i-1,j,k}}\phi_{i-1,j,k}-\frac{m_{i-1,j,k}}{m_{i,j,k}-m_{i-1,j,k}}\phi_{i,j,k},$$

which gives

$$\phi_{i-1,j,k}=\phi_b-\frac{m_{i-1,j,k}}{m_{i,j,k}}\phi_b+\frac{m_{i-1,j,k}}{m_{i,j,k}}\phi_{i,j,k}. \quad (16)$$

Note that though the $(i-1,j,k)$-cell is a ghost cell, the value of $\phi_{i-1,j,k}$ can be computed, at the beginning of the relaxation iterations, from its father (coarser) cell which does exist. Then $\phi_b$ is computed and fixed (because it is the boundary value!). During the subsequent relaxation iterations, the value of $\phi_{i,j,k}$ changes and so does that of $\phi_{i-1,j,k}$ but not $\phi_b$. As we don't have the $(i-1,j,k)$-cell, there is nowhere to store the updated values of $\phi_{i-1,j,k}$, and so we choose to not use it at all, replacing $\phi_{i-1,j,k}$ in the discrete equation by $\phi_b$ and $\phi_{i,j,k}$ using Eq. (16). We then get a boundary-condition (BC)-modified equation

$$\begin{aligned}&\frac{1}{h^2}\left[\phi_{i+1,j,k}+\phi_{i,j+1,k}+\phi_{i,j-1,k}+\phi_{i,j,k+1}+\phi_{i,j,k-1}-\left(6-\frac{m_{i-1,j,k}}{m_{i,j,k}}\right)\phi_{i,j,k}\right]\\&=2\Omega_m a\left(\delta_{i,j,k}-1\right)-\frac{1}{6}\Omega_m a^4\left[(na^2\xi)^{\frac{1}{n+1}}\exp\left(-\frac{u_{i,j,k}}{n+1}\right)-3\left(a^{-3}+4\frac{\Omega_\Lambda}{\Omega_m}\right)\right]-\frac{1}{h^2}\left(1-\frac{m_{i-1,j,k}}{m_{i,j,k}}\right)\phi_b. \quad (17)\end{aligned}$$

Note that because $\phi_b$ is fixed, we could simply move it to the right-hand side of the equation. The BC-modified right-hand side is then only needed to be computed once, before entering the relaxation iterations. There is no need to store either $\phi_b$ or $\phi_{i-1,j,k}$. Note that the left-hand side is also modified and one must be careful here.

[1] If this is a ghost cell then its mask value will be $-1$ but here we try to be general in our description.

#### 2. The Multigrid Implementation

Now consider the multigrid algorithm. For simplicity let us rewrite the above BC-modified equation as

$$L^h\phi^h=f^h \quad (18)$$

in which the superscript means we are on the level with cell size $h$. Note that $L^h$ is a linear operator, which is important here.

Suppose after a number of Gauss-Seidel iterations (pre-smoothing), we get an approximate solution $\hat{\phi}^h$, then

$$L^h\hat{\phi}^h \;=\; \hat{f}^h \;\neq\; f^h \quad (19)$$

in the active cells, and the difference

$$d^h \equiv \hat{f}^h - f^h \tag{20}$$

is called the residual. We could define $\delta\phi^h \equiv \phi^h - \hat{\phi}^h$ and rewrite the equation as

$$L^h \delta\phi^h = -d^h \tag{21}$$

and coarsify this equation as

$$L^H \delta\phi^H = -\mathcal{R} d^h \tag{22}$$

and solve it on the coarser level to accelerate the convergence. Here, the superscript $^H$ indicates that we are on the coarser level on which the cell size is $H = 2h$, and $\mathcal{R}$ is the restriction operator.

To solve this coarsified equation, we need the boundary conditions for $\delta\phi^H$ on the coarser level and, as in Eq. (17), the coarser-level equation will be modified to include correction terms involving $\delta\phi_B$, which is the boundary $\delta\phi$ on the coarser level. However, on the boundary $\delta\phi$ vanishes identically and it turns out that the coarser-level version of Eq. (17) is simpler:

$$\frac{1}{H^2}\left[\phi^H_{i+1,j,k} + \phi^H_{i,j+1,k} + \phi^H_{i,j-1,k} + \phi^H_{i,j,k+1} + \phi^H_{i,j,k-1} - \left(6 - \frac{m^H_{i-1,j,k}}{m^H_{i,j,k}}\right)\phi^H_{i,j,k}\right] = -\mathcal{R} d^h, \tag{23}$$

assuming that the $(i-1, j, k)$ *coarser-level* cell is a ghost cell. Note that the right-hand side of the coarsified equation is *not* BC modified since $\delta\phi_B = 0$, but the left-hand side *is* modified by the lack of neighbouring cells.

Here we have only described the algorithm for two levels, but the generalisation to more levels is trivial.

Finally some comments should be made on the restriction operation $\mathcal{R}d^h$: if a given fine cell is masked[2] then it has no contribution to the coarser-level residual, because the field value in this cell is unmodified by the relaxation and so considered to be exact. At the same time, we shall not restrict residuals into coarse cells which are masked, because this is unnecessary.

#### 3. The Prolongation

Once an approximate coarser-level solution to $\delta\phi^H$ is obtained, say $\hat{\delta\phi}^H$, one can correct the fine-level solution using

$$\hat{\phi}^h \leftarrow \hat{\phi}^h + \mathcal{P}\hat{\delta\phi}^H, \tag{24}$$

where $\mathcal{P}$ is the prolongation operator, to find (expected) more accurate solutions on the fine level.

In the prolongation process, a fine cell receives contribution from its eight neighbouring coarser cells. Of these eight coarse cells:

1. one contains the fine cell and is given a weight of 27/64,
2. three have one common face with the fine cell and are given a weight of 9/64 each,
3. three have one common edge with the fine cell and are given a weight of 3/64 each,
4. one has a common vertex with the fine cell and is given a weight of 1/64.

Of course, these weights sum to unity.

Note that if a given fine cell is masked, then it is outside the active computational domain and corrections are not necessary for it. If the coarse cell is not a valid multigrid cell (it could be a valid AMR cell, though), then its contribution to the fine-cell correction is zero.

### B. The $f(R)$ Equation

Having recalled the multigrid algorithm for the linear (modified) Poisson equation, first presented in [33], we can now have a look at the nonlinear $f(R)$ equation. The nonlinearity introduces certain complications compared to the linear case, which should be taken care of in the numerical implementations.

#### 1. The BC-Modified Equation

As in the linear case, let us suppose that the $(i-1, j, k)$-cell on the finest level is a ghost cell. Then

$$u_{i-1,j,k} = u_b - \frac{m_{i-1,j,k}}{m_{i,j,k}} u_b + \frac{m_{i-1,j,k}}{m_{i,j,k}} u_{i,j,k}. \tag{25}$$

with $u_b$ the boundary value of $u$. Note that this necesarilly means that the equality

$$b_{i-1,j,k} = b_b - \frac{m_{i-1,j,k}}{m_{i,j,k}} b_b + \frac{m_{i-1,j,k}}{m_{i,j,k}} b_{i,j,k} \tag{26}$$

[2] Here "masked" means having a non-positive mask value.

does *not* hold, because $b = e^u$ is nonlinear in $u$ – this point is very important in order to make consistent BC-modified $f(R)$ equation. Instead, we shall use $b_{i-1,j,k} \equiv \exp(u_{i-1,j,k})$ with $u_{i-1,j,k}$ given by Eq. (25).

As before, $u_{i-1,j,k}$ is computed and then used to obtain $u_b$ before entering the relaxation iterations. In the subsequent iterations, $u_{i,j,k}$ is updated and so is $u_{i-1,j,k}$, but not $u_b$.

Removing the $u_{i-1,j,k}$ and $b_{i-1,j,k}$ in Eq. (15) using the above two relations, we arrive at the BC modified $f(R)$ equation:

$$
\begin{aligned}
&\frac{\tilde{c}^2}{2h^2} b_{i,j,k} \left[ u_{i+1,j,k} + u_{i,j+1,k} + u_{i,j-1,k} + u_{i,j,k+1} + u_{i,j,k-1} + \left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b - \left(6 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_{i,j,k} \right] \\
&- \frac{\tilde{c}^2}{2h^2} u_{i,j,k} \left[ b_{i+1,j,k} + b_{i,j+1,k} + b_{i,j-1,k} + b_{i,j,k+1} + b_{i,j,k-1} + \left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) e^{\left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b} e^{\frac{m_{i-1,j,k}}{m_{i,j,k}} u_{i,j,k}} \right] \\
&+ \frac{\tilde{c}^2}{2h^2} \left[ b_{i+1,j,k} u_{i+1,j,k} + b_{i,j+1,k} u_{i,j+1,k} + b_{i,j-1,k} u_{i,j-1,k} + b_{i,j,k+1} u_{i,j,k+1} + b_{i,j,k-1} u_{i,j,k-1} \right] \\
&+ \frac{\tilde{c}^2}{2h^2} \left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b e^{\left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b} e^{\frac{m_{i-1,j,k}}{m_{i,j,k}} u_{i,j,k}} - \Omega_m a (\tilde{\rho}_{i,j,k} - 1) \\
&+ \frac{1}{3} \Omega_m a^4 \left(n a^2 \xi\right)^{\frac{1}{n+1}} \exp\left(-\frac{u_{i,j,k}}{n+1}\right) - \Omega_m a^4 \left(a^{-3} + 4 \frac{\Omega_\Lambda}{\Omega_m}\right) \;=\; 0. \qquad (27)
\end{aligned}
$$

Now we could see clearly where the additional complexity appears. Remember that in the linear case we don't have to store $\phi_b$ because it only appears on the BC-modified right-hand side of the equation. Here, on the other hand, $u_b$ also appears in the term $u_b b_{i,j,k}$ such that its value is needed in each iteration during which $b_{i,j,k}$ is updated. This implies that, for each outermost active cell, we should store $u_b$ for its neighbouring ghost cell(s).

If we define the the left-hand side of Eq. (27) as $\mathcal{L}^h$ for simplicity, then it can be easily shown that

$$
\begin{aligned}
&\frac{\partial \mathcal{L}^h(u_{i,j,k})}{\partial u_{i,j,k}} \\
&= \frac{\tilde{c}^2}{2h^2} b_{i,j,k} \left[ u_{i+1,j,k} + u_{i,j+1,k} + u_{i,j-1,k} + u_{i,j,k+1} + u_{i,j,k-1} + \left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b - \left(6 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_{i,j,k} \right] \\
&- \frac{\tilde{c}^2}{2h^2} \left[ b_{i+1,j,k} + b_{i,j+1,k} + b_{i,j-1,k} + b_{i,j,k+1} + b_{i,j,k-1} + \left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) e^{\left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b} e^{\frac{m_{i-1,j,k}}{m_{i,j,k}} u_{i,j,k}} \right] \\
&- \frac{\tilde{c}^2}{2h^2} \left(6 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) b_{i,j,k} - \frac{\tilde{c}^2}{2h^2} \left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) \frac{m_{i-1,j,k}}{m_{i,j,k}} e^{\left(1 - \frac{m_{i-1,j,k}}{m_{i,j,k}}\right) u_b} e^{\frac{m_{i-1,j,k}}{m_{i,j,k}} u_{i,j,k}} \left(u_{i,j,k} - u_b\right) \\
&- \frac{1}{3(n+1)} \Omega_m a^4 \left(n a^2 \xi\right)^{\frac{1}{n+1}} \exp\left(-\frac{u_{i,j,k}}{n+1}\right). \qquad (28)
\end{aligned}
$$

Note that when $m_{i-1,j,k} = 0$ and $u_b = u_{i-1,j,k}$, the above equations reduce to those for a regular grid with periodic boundary conditions as expected. The relaxation on this level is then done through

$$
u_{i,j,k}^{h,\mathrm{new}} = u_{i,j,k}^{h,\mathrm{old}} - \frac{\mathcal{L}^h\left(u_{i,j,k}^{h,\mathrm{old}}\right)}{\frac{\partial \mathcal{L}^h(u_{i,j,k}^{h,\mathrm{old}})}{\partial u_{i,j,k}^{h,\mathrm{old}}}}. \qquad (29)
$$

### 2. *The Multigrid Implementation*

Let us write the BC-modified $f(R)$ equation as

$$
\mathcal{L}^h\left(u^h\right) = f^h \qquad (30)
$$

on the fine level, where $\mathcal{L}$ is the nonlinear operator acting on $u^h$ defined above. Note that here $f^h = 0$, but we shall still keep it for a reason which will become clear below.

After a number of pre-smoothing iterations, one finds an approximate solution $\hat{u}^h$ on the fine level, which gives

$$
\mathcal{L}^h\left(\hat{u}^h\right) \;=\; \hat{f}^h \;\neq\; f^h. \qquad (31)
$$

Consider the new equation

$$\mathcal{L}^h\left(u^h\right) - \mathcal{L}^h\left(\hat{u}^h\right) \;=\; f^h - \hat{f}^h \;\equiv\; -d^h. \qquad (32)$$

After coarsifying and rearranging, we obtain the coarser-level equation as

$$\mathcal{L}^H\left(u^H\right) = \mathcal{L}^H\left(\mathcal{R}\hat{u}^h\right) - \mathcal{R}d^h. \qquad (33)$$

After a number of relaxation iterations on the coarser level, we find an approximate solution $\hat{u}^H$, and the fine-level solution can be corrected as

$$\hat{u}^{h,\mathrm{new}} = \hat{u}^{h,\mathrm{old}} + \mathcal{P}\left(\hat{u}^H - \mathcal{R}\hat{u}^h\right), \qquad (34)$$

where $\mathcal{P}$ is the prolongation operator.

Different from the linear case, here on the coarser level we are not solving an equation for the correction to the field $u$ (remember in that case we solved $\delta\phi$ on the coarser level) but again $u$ itself. Therefore Eqs. (27, 28) could be applied to the coarse level as well, if we

1. change $h$ to $H$, and
2. find correct coarser-level boundary values for $u$, $u_B$.

The first point is fairly straightforward, while the second needs some further analysis. One must first find the physical boundary on the coarser level which, as explained above, is where the coarser-level mask $m^H$ crosses zero. This is easy because $m^H$ is computed by restricting the fine-cell mask values. The values of $u_B$ in the boundary coarse cells are taken as those in the corresponding AMR cells.

Finally some comments should be made on the restriction operations $\mathcal{R}d^h$ and $\mathcal{R}\hat{u}^h$. $\mathcal{R}d^h$ will be treated similarly as in the linear case for the same reasons explained there.

For $\mathcal{R}\hat{u}^h$, even though a given fine cell is masked it still has contribution, because otherwise the computed $\mathcal{R}\hat{u}^h$ for the coarser cell will be incorrect. But if a coarser cell is masked we will not compute $\mathcal{R}\hat{u}^h$ for it, since this will be unused anyway, as in the case of $\mathcal{R}d^h$.

The truncation error $\tau^h$ can be estimated as

$$\tau^h \approx \mathcal{L}^H\left(\mathcal{R}\hat{u}^h\right) - \mathcal{R}\mathcal{L}^h\left(\hat{u}^h\right), \qquad (35)$$

and similarly for other levels. This could provide a stopping (convergence) criterion for the multigrid iteration, which in our case is

$$\left|d^h\right| \lesssim \alpha\left|\tau^h\right|, \qquad (36)$$

where $\alpha \leq 1/3$ is a predefined constant.

#### 3. The Prolongation

As mentioned above, in the nonlinear case we have to prolongate the quantity $(\hat{u}^H - \mathcal{R}\hat{u}^h)$, but not the residual itself. The prolongated result can then be used to correct the fine-level solutions.

The prolongation here is done in the same way as that for the residual of the linear (modified) Poisson equation, to be consistent. Details shall not be presented here.

## VI. CODE TESTS

In this section we shall give the results of some tests we have performed to show that the above algorithm and implementation work correctly and efficiently.

### A. $f(R)$ Equation Solver On Domain Grid

The most important part in the modified `RAMSES` code, which is the topic of the last two sections, is the equation of motion for $f(R)$ gravity or the new degree of freedom(s) in other theories. Here we have performed a range of tests for it with different configurations of the matter density field.

#### 1. Homogeneous Matter Density Field

In a universe with homogeneous density, we know that the quantity $f_R$ is homogeneous and it exactly takes its background value $\bar{f}_R$, namely

$$\bar{\tilde{f}}_R = -\frac{na^2\xi}{\left[3\left(a^{-3} + 4\frac{\Omega_\Lambda}{\Omega_m}\right)\right]^{n+1}}, \qquad (37)$$

everywhere. Therefore, as the simplest test of the $f(R)$ equation solver, one has to show that in such a homogeneous field, given some random guess of $\tilde{f}_R$ on the cells of the simulation mesh, after a reasonable number of Gauss-Seidel relaxation sweeps, the solution converges to the above background value. Such simple test have been used previously in [26, 27] to show that the `MLAPM` solver for extra degrees of freedom works well.

We have performed this test for $a \approx 0.04, n = 1$ and a value of $\xi$ corresponding to $|f_{R0}| = 10^{-5}$. The result is shown in Fig. 1, where we plot the values of $\tilde{u} = \log(-\tilde{f}_R)$ in the cells in $x$-direction, before and after the Gauss-Seidel relaxation, for two different random initial guesses. We can see that the final solution agrees with the analytical result (the horizontal line) very well (see figure caption for more details).

#### 2. Point Mass

As a second test of the `RAMSES` $f(R)$ equation solver, let us consider the solution of $f_R$ around a point mass at the origin, for which case we have an analytical solution which is accurate except for the regions very close to the mass. Such a test has been used previously in [15, 26].

Following [15], we construct the point-mass density field as

$$\delta_{i,j,k} = \begin{cases} 10^{-4}\left(N^3 - 1\right), & i = j = k = 0; \\ -10^{-4}, & \text{otherwise.} \end{cases} \qquad (38)$$

in which $i, j, k$ are respectively the cell indices in $x, y, z$ direction. In the test we have used a cubic box with size

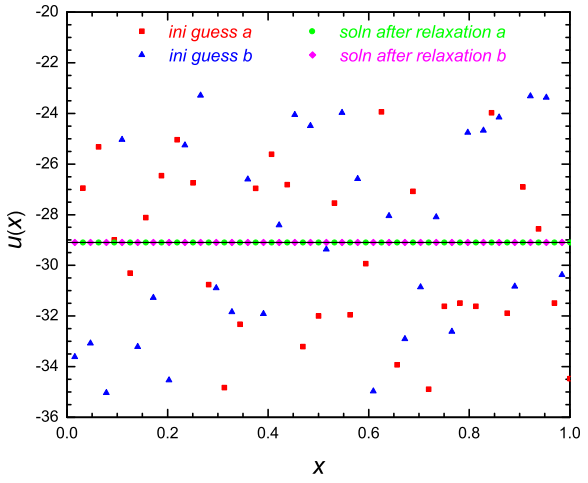


FIG. 1. (Colour online) Test of the solver for the $f(R)$ equation in a constant matter density field. Only results in the cells along the $x$-axis are shown, and the $x$-coordinate is rescaled by the size of the simulation box so that $x \in [0,1]$. Two initial guesses of $\tilde{u} \equiv \log(-\tilde{f}_R)$ have been tried (the red squares and blue triangles), the final answer corresponding to which are respectively denoted by green filled circles and purple diamonds. The black horizontal line is the exact solution. Note that the results with the first initial guess (filled squares and circles) have been shifted rightwards to make the plot clearer.

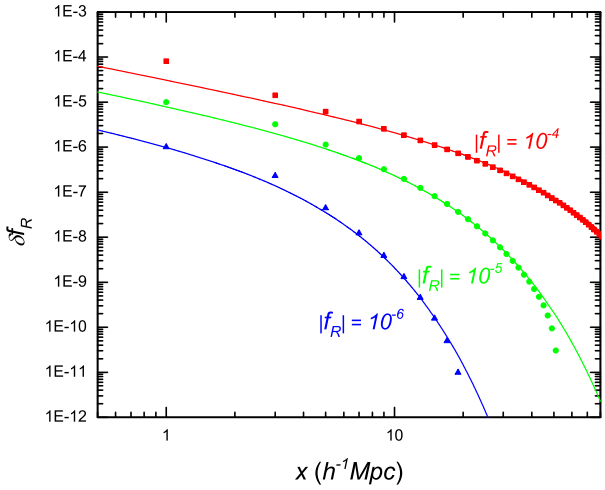


FIG. 2. (Colour online) The solution to $\delta f_R \equiv f_R - \bar{f}_R$ around a pointed mass constructed according to Eq. (38), for three models with $|f_{R0}| = 10^{-6}$ (blue triangles), $10^{-5}$ (green filled circles) and $10^{-4}$ (red squares) respectively. The solid curves are the corresponding analytical approximations which are accurately far from the point mass. Only the solutions along the $x$-axis are shown.

$256h^{-1}$Mpc and 128 grid cells in each direction. The other physical parameters are chosen as $a = 1, n = 1$ and we have used three values of $\xi$ corresponding to $|f_{R0}| = 10^{-6}, 10^{-5}, 10^{-4}$.

Meanwhile, the analytical solution can be obtained approximately by solving the equation

$$\nabla^2 \delta f_R \approx m_{\rm eff}^2 \delta f_R \tag{39}$$

in which the effective mass of the scalar degree of freedom $\delta f_R$, $m_{\rm eff}$, is given by

$$m_{\rm eff} = \frac{1}{\sqrt{3}} \frac{1}{n(n+1)\xi} \left[ 3\left(1 + 4\frac{\Omega_\Lambda}{\Omega_m}\right)\right]^{n+2} \Omega_m H_0^2 . \tag{40}$$

Fig. 2 shows the comparison between the numerical solutions to $\delta f_R$ along the $x$-axis (symbols) and analytical solutions (solid curves), and we can see that the two agree very well for all three models used in this test.

### 3. *Sine Density Field*

As our third test, let us consider the sine density field introduced in [15], which (after some modification to account for the code units) is given by

$$\delta(x) = \frac{\tilde{c}^2}{\Omega_m a} (2\pi)^2 \frac{n a^2 \xi}{\left[3\left(a^{-3} + 4\frac{\Omega_\Lambda}{\Omega_m}\right)\right]^{n+1}} \sin(2\pi x) + \left(1 + 4a^3 \frac{\Omega_\Lambda}{\Omega_m}\right) \left\{ [2 - \sin(2\pi x)]^{-\frac{1}{n+1}} - 1 \right\} \tag{41}$$

in which $x$ is rescaled such that $x \in [0,1]$. Notice that we consider only $x$-dependence, which is equivalent to a one-dimensional configuration. The solution to this density field can be analytically worked out to be,

$$\tilde{f}_R(x) = \frac{n a^2 \xi}{\left[3\left(a^{-3} + 4\frac{\Omega_\Lambda}{\Omega_m}\right)\right]^{n+1}} \left[\sin(2\pi x) - 2\right] . \tag{42}$$

Fig. 3 shows the test results for the sine density field given above, with $a = n = 1$ and three values of $\xi$ corresponding to $|f_{R0}| = 10^{-6}, 10^{-5}, 10^{-4}$ respectively. It can be seen that the numerical solutions (symbols) agree with the analytical solutions (solid curves) very well.

### 4. *Gaussian Density Field*

The last test of the $f(R)$ equation solver on the domain grid uses a Gaussian type density configuration. Again, here we only consider one-dimensional case, where the

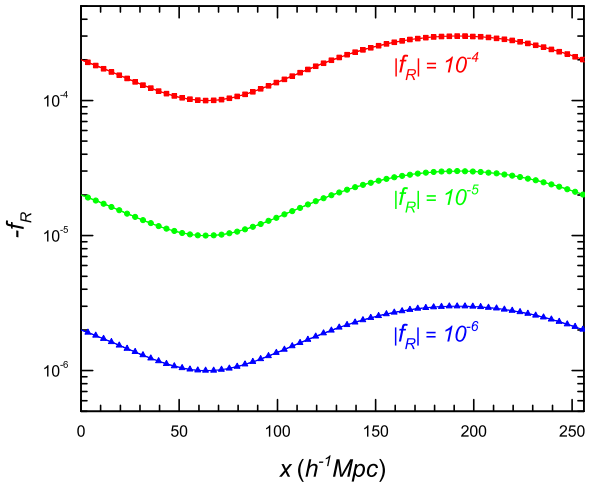


FIG. 3. (Colour online) Solutions of $f_R$ in a one-dimensional ($x$-direction) sine density field constructed using Eq. (41), for three models with $|f_{R0}| = 10^{-6}$ (blue triangles), $10^{-5}$ (green filled circles) and $10^{-4}$ (red squares) respectively. The solid curves are the corresponding analytical results. A simulation box with side length of $256h^{-1}$Mpc and 256 grid cells on each side is used in the computation. $x$ is in physical units.

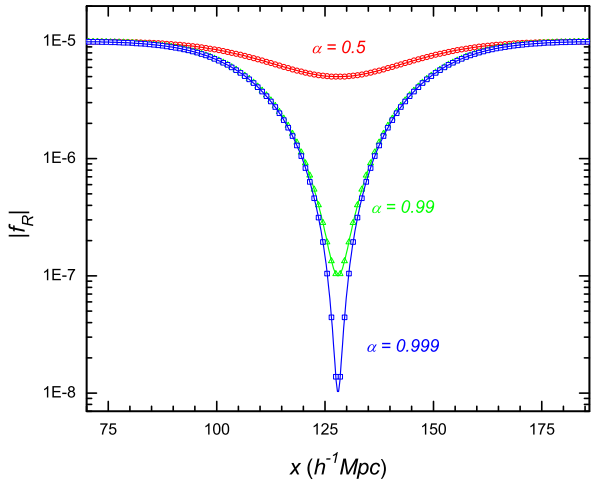


FIG. 4. (Colour online) Solutions of $f_R$ in a one-dimensional ($x$-direction) Gaussian-type density configuration constructed using Eq. (43), for the models with $|f_{R0}| = 10^{-5}$ and $\alpha = 0.5$ (red circles), 0.99 (green triangles) and 0.999 (blue squares). The solid curves are the corresponding analytical results from Eq. (45). A simulation box with side length of $256h^{-1}$Mpc and 256 grid cells on each side is used in the computation and the $f(R)$ equation is only solved on the regular domain grid. $x$ is in physical units.

density field is specified by

$$\delta(x) = \frac{c^2}{\Omega_m a}\frac{2A\alpha}{W^2}\exp\left[-\frac{(x-0.5)^2}{W^2}\right]\left[1-2\frac{(x-0.5)^2}{W^2}\right] + \frac{1}{3}a^3\left\{\frac{na^2\xi}{A\left[1-\alpha\exp\left(-\frac{(x-0.5)^2}{W^2}\right)\right]}\right\}^{\frac{1}{n+1}} - \left[1+4\frac{\Omega_\Lambda}{\Omega_m}a^3\right] \quad (43)$$

where again $x$ has been scaled to code units so that $x \in [0,1]$, $W$, $\alpha$ are numerical constants which respectively specify the width and height of the density field, which obviously peaks at $x = 0.5$, and $A$ is defined by

$$A \equiv \frac{na^2\xi}{\left[3\left(a^{-3}+4\frac{\Omega_\Lambda}{\Omega_m}\right)\right]^{\frac{1}{n+1}}} \quad (44)$$

Note that such a density field is not exactly periodic at the edges of the simulation box, but given that $W$ is small enough, $\delta \to 0$ at the box edges and periodic boundary conditions are approximately satisfied.

The solution to $f_R$ can then be obtained analytically and is

$$f_R = -A\left[1-\alpha\exp\left(-\frac{(x-0.5)^2}{W^2}\right)\right], \quad (45)$$

which clearly shows that when $\alpha \to 1$ $|f_R|$ could be made very small at $x = 0.5$ while at $x \to 0$ or $x \to 1$ $f_R$ goes to its background value.

We have implemented Eq. (43) into our numerical code and the numerical solutions for $f_R$ are shown in Fig. 4. For simplicity we only show the results for $|f_{R0}| = 10^{-5}$, but for three different values of $\alpha = 0.5, 0.99$ and 0.999 (the open circles, open triangles and open squares in Fig. 4 respectively). We can see that the numerical results agree with the analytical solution Eq. (45) very well.

### B. $f(R)$ Equation Solver On Refinements

The above four tests show that our solver for the $f(R)$ equation actually works accurately on the regular domain grid, but this is not sufficient for the code test since the $f(R)$ equation is also solved on refinements where, as we have seen, the equations take different forms due the complex treatment of the boundary conditions. It is therefore essential to test the $f(R)$ equation solver on the refinements as well, as we shall do in this subsection.

#### 1. *Two-level Gaussian Density Field*

The Gaussian-type density configuration could provide a good check of the multilevel $f(R)$ equation solver because the density peak can be made arbitrarily high by adjusting the parameter $\alpha$. Near its peak, the density field changes rapidly and a higher spatial resolu-

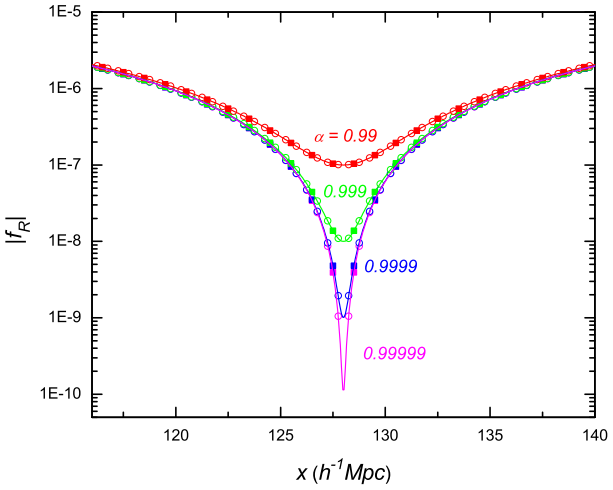


FIG. 5. (Colour online) The same as Fig. 4, but for the models with $|f_{R0}| = 10^{-5}$ and $\alpha = 0.99, 0.999, 0.9999, 0.99999$ (from top to bottom: red, green, blue, magenta). The $f(R)$ equation is solved on two levels: level 8 (the regular domain grid) and level 9 (the first refinement), and their numerical solutions are represented by filled squares and empty squares respectively. The solid curves are the corresponding analytical results from Eq. (45).

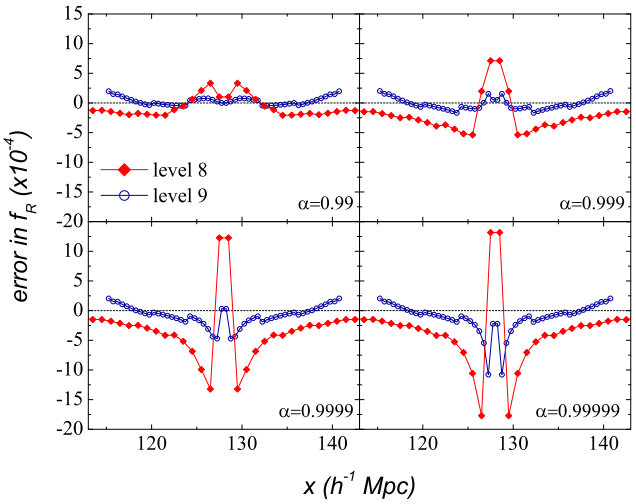


FIG. 6. (Colour online) Numerical errors of $f_R$ (defined as the fractional difference between the numerical result and the exact solution Eq. (45)) for the models with $|f_{R0}| = 10^{-5}$ and $\alpha = 0.99$ (upper left panel), 0.999 (upper right), 0.9999 (lower left) and 0.99999 (lower right). The $f(R)$ equation is solved on two levels: level 8 (the regular domain grid, red filled diamonds) and level 9 (the first refinement, blue hollow circles).

tion is needed to compute $f_R$ accurately. Consider the case where the regular domain grid is refined only once, in regions where the density value exceeds some certain threshold (we shall call this a two-level problem, and in the present numeric example the coarse and fine levels are respectively levels 8 and 9). The density values in both the coarse and refined cells are given by Eq. (43), while the values of $f_R$ at the fine-level boundaries are computed from interpolation of those in the nearby coarse-level cells, as discussed above.

Fig. 5 shows the numerical values of $f_R$ on both levels in the region covered by the refinement. We show the results for four different values of $\alpha$ (0.99, 0.999, 0.9999, 0.99999 from top to bottom), and for each $\alpha$ the results from the coarse and fine levels are denoted respectively by filled squares and empty circles. For comparison we have also plotted the analytical results Eq. (45) as solid curves. As we can see, both fine-level and coarse-level results are virtually indistinguishable from the exact solution by eye.

This does not mean that the refinement is unnecessary however, because, as shown in Fig. 5, the fine level has more data points and could probe regions closer to the extreme value of $f_R$, which corresponds to the high-density region where high resolution is needed.

To see more quantitatively how well the solutions from different levels agree with the exact results, we have plotted in Fig. 6 the percentage errors of the numerical solutions for the same four models considered in Fig. 5. The following features could be easily observed from this plot:

1. At the coarse level (level 8), the numerical error increases as the density fluctuation becomes stronger (i.e., $\alpha \to 1$), which is as expected;

2. Around the peak of the density field, the fine level (level 9) consistently gives more accurate numerical results than level 8 for all 4 models, thanks to its higher spatial resolution;

3. The improvement of level-9 result over that of level-8 is smaller as $\alpha \to 1$, because at the peak of the density field even level 9 will not be very accurate;

4. The level-9 result is actually less accurate at the refinement boundaries, because of the numerical error that comes from setting boundary conditions using interpolation. Furthermore, the error at refinement boundaries is the same for all models, at $\sim 0.025\%$, which is negligible anyway. Note that we could improve on this using a higher-order interpolation scheme at boundaries, which will be considered in future work.

### 2. *Three-level Gaussian Density Field*

In the above we have performed a test using the domain grid and one refinement level, but the generalisation to more refinement levels is very straightforward (and indeed done automatically in the code, with the refinement

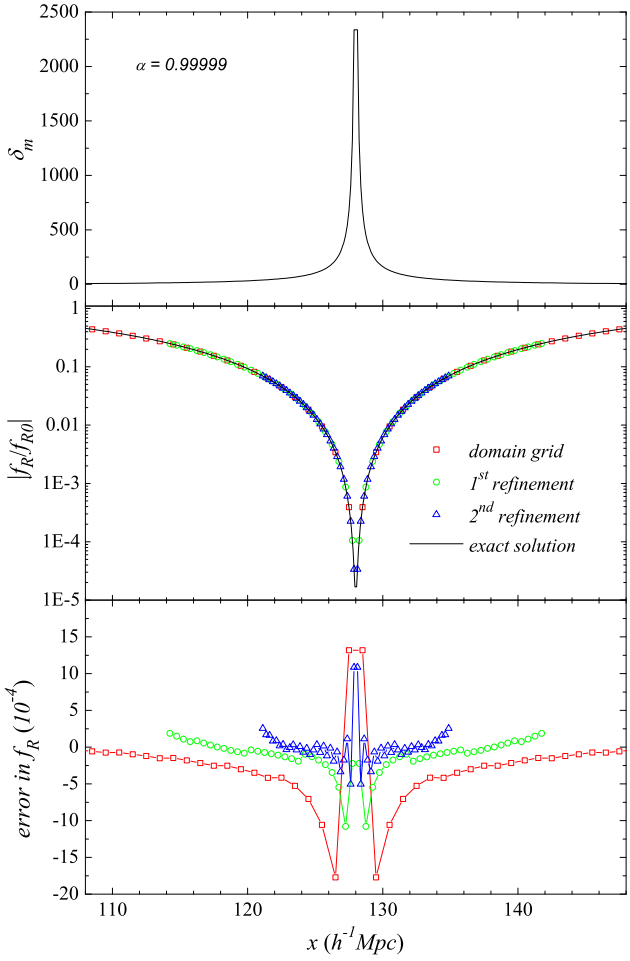


FIG. 7. (Colour online) *Upper Panel:* The one-dimensional matter density field as a function of the $x$ coordinate (in physical units), to provide an impression about the height of its peak. $\delta_m \equiv \rho_m/\bar{\rho}_m - 1$. *Middle Panel:* The same as Fig. 5, but for the model with $|f_{R0}| = 10^{-5}$ and $\alpha = 0.99999$ only for simplicity. The $f(R)$ equation is solved on three levels: level 8 (the regular domain grid) and level 9 (the first refinement) and level 10 (the second refinement), and their numerical solutions are represented by red squares, green circles and blue triangles respectively. The black solid curve is the corresponding analytical result from Eq. (45). *Lower Panel:* The same as Fig. 6, but for the model studied here; the use of symbols is the same as in middle panel.

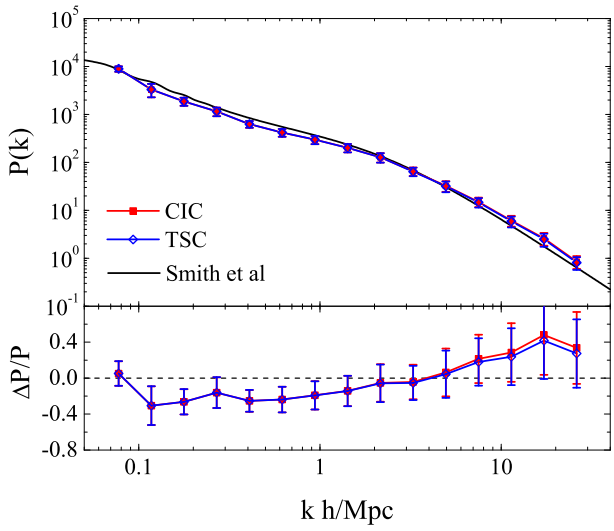


FIG. 8. (Colour online) *Upper Panel:* Comparison of the nonlinear matter power spectra using the TSC (blue diamonds) and CIC (red squares) density assignment schemes to that obtained using Smith *et al.* fit (black solid curve). *Lower Panel:* The relative differences of the TSC and CIC power spectra with respect to that of the Smith *et al.* fit. The dashed black line is zero identically. The cosmological simulations here use a cubic box with size $100h^{-1}$Mpc and 256 cells on each side of the domain grid. The error bars show the sample variance over 5 different realisations. All results are at redshift 0.

criterion specified *a priori*). Here we only give an example with three refinement levels, again making use of the above Gaussian-type density field.

Fig. 7 shows the results of the three-level test. In the upper panel we have plotted the density field $\delta_m$ as function of $x$, and it could be seen that it has a rather high and sharp peak near $x = 128h^{-1}$Mpc, which is exactly where higher spatial resolution and thus refinements are needed. In the middle panel we have shown the numerical solutions from the three refinement levels (8, 9 and 10) in the spatial regions covered by them, and again we see that the agreements with the exact solution are very good (indistinguishable by eye). The numerical errors from the $f(R)$ equation solver at the three levels are shown in the lower panel, and we see the same patterns as observed in Fig. 6, namely near the density peak higher refinement level produces smaller error, but at the refinement edges slightly bigger error occurs because of the interpolation used in setting the boundary conditions. The magnitude of the latter error source, however, is much smaller than that near the density peak, and is negligible for cosmological simulations anyway.

These tests indicate that our $f(R)$ equation solver is reasonably accurate and therefore reliable also on the refinement(s), which is one of the most significant achievements of the present code.

### C. Density Assignment and Force Interpolation

The default `RAMSES` code uses the CIC (cloud-in-cell) scheme to do density assignment and force interpolation. While the CIC scheme works well and is the most widely used due to a compromise of accuracy and cost, the TSC (triangle-shaped clouds) scheme actually produces a smoother density field which more resembles the true dark matter distribution, especially when the parti-

TABLE I. A brief summary of the key features of the three known grid-based codes for $N$-body simulations in modified gravity and/or dynamical dark energy theories. Here `OC` stands for *Oyaizu Code*, and by `MLAPM` we mean the modified version to include the multigrid solver for the extra degree(s) of freedom.

| Codes | `OC` | `MLAPM` | `ECOSMOG` |
|---|---|---|---|
| Reference | Ref. [15] | Refs. [20, 23] | This paper |
| Density assignment scheme | CIC | TSC | TSC |
| Force interpolation scheme | CIC | TSC | TSC |
| Parallelisation | `OpenMP` | N/A | `MPI` |
| Adaptive refinement? | No | Yes | Yes |
| Multigrid on regular grid? | Yes | Yes | Yes |
| Multigrid relaxation arrangement on regular grid | V-cycle | V-cycle/adaptive cycle | V-cycle |
| Multigrid on refinement? | N/A | No | Yes |
| Multigrid relaxation arrangement on refinement | N/A | N/A | V-cycle |
| Programming language | `C++` | `C` | `FORTRAN 90` |

cle number is not large enough. Furthermore, this results in a more isotropic force around a point mass, a desirable property. For this reason, we have added a new routine in the code to do the TSC density assignment. Correspondingly, the force interpolation also needs to use the TSC scheme to ensure momentum conservation. For more discussions on the different density assignment schemes we refer the readers to [31].

Because in the TSC density assignment scheme a particle's cloud [3] spreads more than in the CIC scheme, the resulted density field will be smoother and as a result we expect less small-scale structure from $N$-body simulations. This is confirmed by our numerical results shown in Fig. 8, where we have plotted the nonlinear matter power spectra from our ΛCDM runs using TSC and CIC respectively, and compared them with the Smith *et al.* fit [32]. We can see that on very large scales (small $k$) TSC gives virtually the same prediction as CIC, but on smaller scales (large $k$) it produces less power than the latter. On the other hand, the difference between TSC and CIC (a few percent) is much smaller than their difference from the Smith *et al.* fit (up to 40% on small scales).

A simpler test of the code segments for the TSC density assignment is to check that the average density of all grid cells on the domain grid is 1.0. We have monitored this at each time step of all our cosmological simulations and found that this is indeed the case. As a result, we believe that this part of our `ECOSMOG` code is reliable.

[3] In density assignment, a particle only contributes to the density field within a finite region around it, and this is called that particle's cloud. The particle's contribution to density field integrated over its cloud is its mass. As a result, the bigger the cloud is, the smoother the resulted density field will be.

### D. Performance Tests

In addition to a more accurate $f(R)$ equation solver, another aim we want to achieve for our code is efficiency. The efficiency in the $f(R)$ equation solver can be greatly improved by two factors. The first (as mentioned earlier) is to use multigrid rather than a single grid in arranging the Gauss-Seidel relaxation to speed up its convergence, and the other is using parallelisation to take advantage of the supercomputing resources. These two elements are incorporated in the original `RAMSES` code for the Poisson solver, and in our code we apply them to the $f(R)$ equation solver as well.

#### *1. Performance of V-cycle*

We use a V-cycle to arrange the multigrid relaxation, as in the modified `MLAPM` code [20, 23]. As a check of the improvement that is achieved in this way, we have plotted in Fig. 9 the convergence rates for three models (details of which can be found in the figure caption) using single-grid relaxation (solid curves) and multigrid V-cycle (dashed curves). The convergence rate is equivalent to the rate at which the module of the residual decreases. We can see that with the multigrid relaxation the residual drops below $10^{-12}$ after tens of Gauss-Seidel sweeps on the finest grid, while with the single-grid relaxation this depends sensitively on both the model and its parameters – in the worst case we have tested here (the sine density field with $f_{R0} = -10^{-4}$, solid curve with filled squares in Fig. 9) the residual is still bigger than $10^{-4}$ even after 1000 Gauss-Seidel sweeps![4] Clearly

[4] Of course this also depends on the initial guess – if it is close enough to the true solution then rapid convergence could be expected. But in most situations we have no idea about the exact solution and cannot find such good initial guesses.

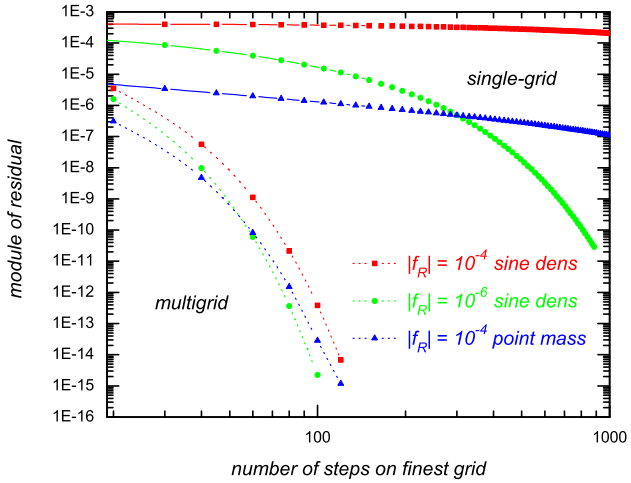


FIG. 9. (Colour online) The reduction of the residual for the $f(R)$ equation for three models: $|f_{R0}| = 10^{-4}$ with a sine-type density field as discussed above (red squares), $|f_{R0}| = 10^{-6}$ with a sine-type density field (green circles) and $|f_{R0}| = 10^{-4}$ for a point mass discussed above (blue triangles). The results using a single grid are represented by symbols connected by solid curves, while those using the multigrid method (arranged by V-cycles) by symbols connected by dashed curves.

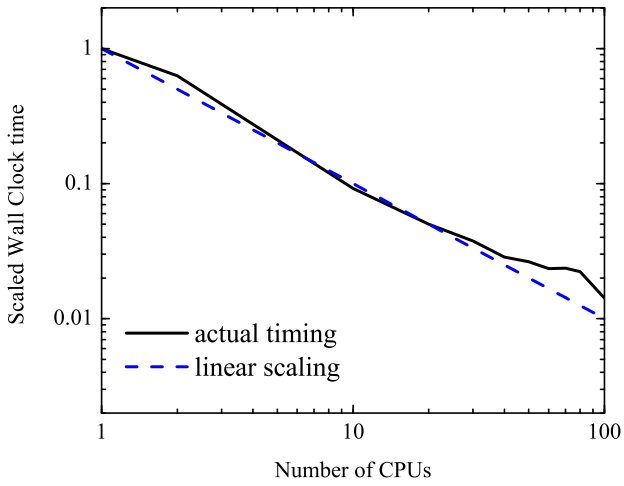


FIG. 10. (Colour online) The wallclock time of the $f(R)$ equation solver as a function of the number of CPUs used in the parallelised computation. The actual measurement is shown as the black solid curve, while the blue dashed line is a linear scaling just for comparison. It is obvious that the time-CPU number scaling is approximately linear for CPU numbers up to $\sim 100$.

the single-grid relaxation is impractical in most realistic simulations.

One difference between the code here and the modified `MLAPM` [20, 23] is that here the multigrid relaxation method is used not only on the regular domain grid, but also on the refinements. In contrast, in `MLAPM` single-grid relaxation is used on refinements, which gains ease in the code implementation but at the cost of computing time and accuracy. Tests of our code show that on the refinements the V-cycle could bring the residual down to $10^{-12}$ within a reasonable number of Gauss-Seidel sweeps, just as it does on the regular grid. Therefore, we expect more accurate results than those obtained in previous works; we will come back to this in a future work.

#### 2. *Parallelisation Performance*

Parallelisation is a way to let many CPUs share the computational overhead and therefore reduce the total physical time needed to finish the job. It has greatly improved the efficiency of numerical simulations in cosmology. Our parallelisation here uses the structure of the original `RAMSES` code, which is parallelised using `MPI` and has been shown to work very well for the Poisson equation solver. This is made possible by the fact that the mesh structures used for the Poisson and $f(R)$ equations are exactly the same, and so the structures for communication (which is a crucial ingredient of parallelisation) could be shared by the Poisson and $f(R)$ equation solvers, provided that they will not be modified until both equations are solved and relevant quantities appropriately stored.

To check the efficiency of the parallelisation, we have used different numbers of CPUs to solve the $f(R)$ equation (only once) and recorded the wallclock times used. This is plotted in Fig. 10 from which we can see that, at least when the CPU number ranges between 1 and $\sim 100$, the wallclock time roughly scales linearly with the number of CPUs. This shows that the parallelisation does help to make the code faster, and therefore most suitable for future simulations with large numbers of particles, large box sizes and high spatial resolutions.

### E. Cosmological Simulations

The aim of the modified code is to run cosmological simulations for modified gravity and dynamical dark energy theories, $f(R)$ gravity as an example. Therefore, we still need to test its reliability in such simulations, which is the purpose of this subsection.

As one cosmological test, we have performed simulations for the $f(R)$ gravity model with $|f_{R0}| = 10^{-5}$ using a $100h^{-1}$Mpc simulation box and measured their matter power spectra. In order to reduce the cosmic variance, we show the matter spectrum for $f(R)$ gravity rescaled by that for a $\Lambda$CDM model simulated using the same initial condition. To reduce the sample variance, we make bins uniform in logarithm of the wavenumber $k$.

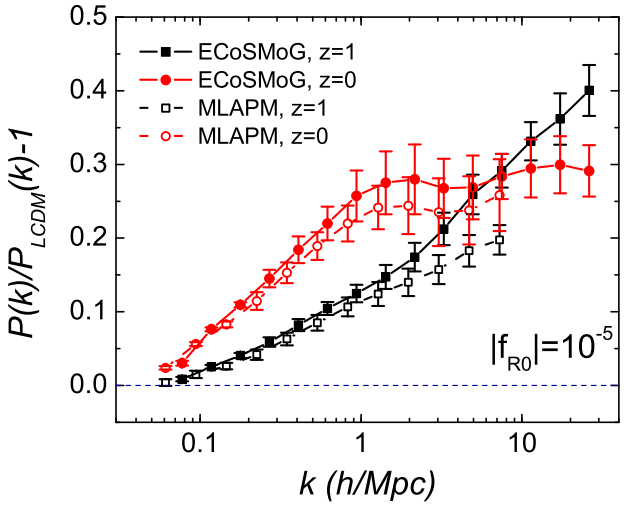


FIG. 11. (Colour online) Nonlinear power spectra at redshifts 1 (black filled squares) and 0 (red filled circles), from cosmological simulations for the $f(R)$ model with $|f_{R0}| = 10^{-5}$. The horizontal axis is the wave number in the unit of $h\mathrm{Mpc}^{-1}$, and the vertical axis is the fractional difference between the $f(R)$ and $\Lambda$CDM results. We have considered five realisations for each model, using the same set of initial conditions, box size ($100h^{-1}$Mpc), spatial resolution (256 cells on each side for the domain grid) and refinement criterion. The power spectra are averaged and the standard deviations are shown as error bars. We have also binned the horizontal axis to smooth the result [22]. The corresponding results from [22] have been plotted as black empty squares and red empty circles for comparison.

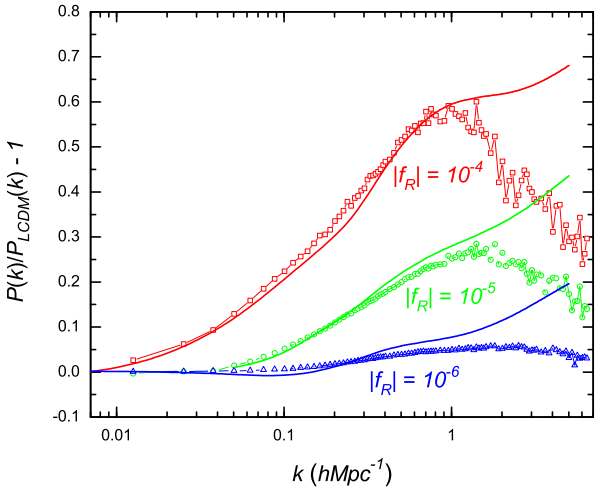


FIG. 12. (Colour online) Nonlinear matter power spectra at redshift 0, from cosmological simulations for the $f(R)$ models with $|f_{R0}| = 10^{-4}$ (red empty squares), $10^{-5}$ (green empty circles) and $10^{-6}$ (blue empty triangles). The horizontal axis is the wave number in the unit of $h\mathrm{Mpc}^{-1}$, and the vertical axis is the fractional difference between the $f(R)$ and $\Lambda$CDM results. The solid curves are the corresponding results from Smith *et al.* fit.

The numerical results are summarised in Fig. 11 (more technical details are described in the caption). To make comparison with the $\Lambda$CDM result, displayed in Fig. 11 are the relative differences between the $f(R)$ and $\Lambda$CDM power spectra. At redshift $z = 1$ ($a = 0.5$), the fractional difference monotonically increases towards smaller scales, which have already been within the reach of the fifth force and started experiencing the boosted matter clustering; at redshift 0 ($a = 1.0$) the fractional difference goes down at smaller scales, because the particles have been accelerated, which helps to erase the small-scale structure to a certain degree [22].

We have also compared the results from `ECOSMOG` and those from the modified `MLAPM` code (empty symbols) in Fig. 11, which shows that the two codes agree quite well on large scales. On smaller scales, the quantitative agreement becomes worse but the qualitative features are still the same. In particular, the results at $z = 0.0$ are within the $1\sigma$ error bars of each other. The discrepancy on small scales comes from the fact that the `ECOSMOG` code could achieve much higher accuracy than `MLAPM`: on the domain grid (with 256 cells in our `ECOSMOG` runs and 128 cells in the `MLAPM` runs) the convergence criterion is $|d^h| < 10^{-12}$ for the former code and $|d^h| < \max\left(10^{-6}, 0.03|\tau^h|\right)$ for the latter; on the refinements it becomes $|d^h| < 10^{-8}$ and $|d^h| < \max\left(10^{-6}, 0.03|\tau^h|\right)$ for the two codes[5].

Because $|f_R| \ll 1$ appears in the denominator of the right-hand side of the Poisson equation, a small difference in $f_R$ can be magnified and result in a big difference in the solution of the Newtonian potential. This is why a high accuracy in the solution of $f_R$ is important, and can also explain why the difference between the `ECOSMOG` and `MLAPM` results in Fig. 11 is bigger at earlier times (when $|f_R|$ is smaller and its accuracy more important). Indeed, we find the small-scale structure in the $f(R)$ gravity to be quite sensitive to the convergence criterion of the $f(R)$ equation solver as well as the size of the simulation box, the refinement criterion *etc.*, and a thorough study needs to be conducted to clarify these issues before systematic simulations of the $f(R)$ gravity are performed. We leave this as a future work.

As the other cosmological test, we have also run simulations for three $f(R)$ gravity models with $|f_{R0}| = 10^{-6}$, $10^{-5}$ and $10^{-4}$ respectively, for a bigger simulation box ($500h^{-1}$Mpc now). Because the measured matter power

[5] We note that on refinements it is almost always the case that $10^{-6} \ll 0.03|\tau^h|$, because the truncation error can be quite big on the fine levels. This means that the `MLAPM` $f(R)$ equation solver was deemed as having converged even when $|d^h| \sim \mathcal{O}(0.01 - 1)$ on the higher refinements, while the `ECOSMOG` solver has a much more stringent criterion of $|d^h| < 10^{-8}$, and naturally will give more accurate results. Without using multigird relaxation on the refinements, such an accuracy will not become practical.

spectra are much smoother than the case above, here we shall not average over realisations or perform the binning. Fig. 12 shows the enhancements of the power spectra in these models (symbols), and also the results using Smith *et al.* fit (dashed curves). We could see that on the large scales they agree reasonably (for $k \ll 1h\mathrm{Mpc}^{-1}$ the difference is at most a few percent). This serves as another check of the consistency and reliability of our code.

As a final note, we find that simulations using bigger boxes generally takes less time to complete, provided that the particle numbers are the same. This is because the clustering is less developed and the cell size is larger, so that smaller scales are not resolved as well as for smaller box sizes. As a consequence, time steps are larger, and there are few refinement levels, even with the same refinement criterion. The density field is smoother, which makes the $f(R)$ equation solver converge more quickly. With 80 CPUs, our simulations with $500h^{-1}$Mpc box size can finish within a couple of hours, in contrast to about 24 hours for the simulations with $100h^{-1}$Mpc box size.

## VII. SUMMARY AND CONCLUSIONS

Large $N$-body simulations for modified gravity or dynamical dark energy theories have become more and more important in the wake of the inflow of high-quality observational data in the near future. To best extract information from these data, one needs better theoretical understandings of different models, in particular on scales of galaxy cluster and smaller sizes, where the commonly-used linear perturbation analyses are no longer valid.

In this work we have presented a new, efficiently parallelised, accurate and fast code called `ECOSMOG` based on the public code `RAMSES`. The `ECOSMOG` code is specifically designed for performing large $N$-body and hydrodynamic simulations for modified gravity and generic dark energy theories with additional scalar degree(s) of freedom. We have clarified a number of numerical and technical issues about implementing the (usually nonlinear) equations of motion for those new degrees of freedom, and preformed a series of code tests, using $f(R)$ models, for the issues of the accuracy of the multigrid scalar field solver on the domain grid, on the refinements, the density assignment and the force interpolation scheme, as well as the efficiencies of the scalar field solver and of the parallelisation. Our results show that `ECOSMOG` works extremely well.

`ECOSMOG` closely follows the convention and style of the original `RAMSES` code. Special efforts have been made to not mess the default code, and the scalar field solver has been designed as several additional `F90` files which are maximally parallel to the `RAMSES` Poisson solver. There are some modifications to the default code, mainly to ensure smooth interfaces (such as make sure useful quantities are not destroyed between the calls to the scalar field and Poisson solvers), but these have been kept to minimal level.

`ECOSMOG` is easy to use and modify for other gravity or dark energy models, making it a potentially powerful tool for massive $N$-body and hydrodynamical simulations for interesting theories of gravity and dynamical dark energy.

## ACKNOWLEDGMENTS

BL is supported by Queens' College and the Department of Applied Mathematics and Theoretical Physics of University of Cambridge. GBZ and KK are supported by STFC grant ST/H002774/1. KK also acknowledges support from the European Research Council and the Leverhulme Trust. BL thanks Drs. David Mota and Hongsheng Zhao for encouragement to do this work, and the Institute of Cosmology and Gravitation for its host when this work was initialised. The test runs discussed in this paper have been performed on the `SCIAMA` supercomputer of University of Portsmouth; the initial conditions for the cosmological runs were generated using the `MPgrafic` code [34] and the matter power spectra were measured using `POWMES` [35].

[1] E. J. Copeland, M. Sami and S. Tsujikawa, Int. J. Mod. Phys. D, **15**, 1753 (2006).
[2] I. Zlatev, L. M. Wang and P. J. Steinhardt, Phys. Rev. Lett., **82**, 896 (1999).
[3] C. Armendariz-Picon, V. Mokhanov and P. J. Steinhardt, Phys. Rev. Lett., **85**, 854438 (2000).
[4] L. Amendola, Phys. Rev. D, **62**, 043511 (2000).
[5] J. Khoury and A. Weltman, Phys. Rev. D, **69**, 044026 (2004).
[6] D. F. Mota and D. J. Shaw, Phys. Rev. D, **75**, 063501 (2007).
[7] K. Hinterbichler and J. Khoury, Phys. Rev. Lett., **104**, 231301 (2010).
[8] , S. M. Carroll, A. De Felice, V. Duvvuri, D. A. Easson, M. Trodden and M. S. Turner, Phys. Rev. D, **71**, 063513 (2005).
[9] F. Perrotta and C. Baccigalupi, Phys. Rev. D, **61**, 023507 (1999).
[10] G. Dvali, G. Gabadadze and M. Porrati, Phys. Lett. **B485**, 208 (2000).
[11] K. Koyama, Gen. Rel. Grav., **40**, 421 (2008).
[12] R. Durrer and R. Maartens, Gen. Rel. Grav., **40**, 301 (2008).
[13] T. P. Sotiriou and V. Faraoni (2008), arXiv:0805.1726.
[14] B. Li and J. D. Barrow, Phys. Rev. D **75**, 084010 (2007).
[15] H. Oyaizu, Phys. Rev. D**78**, 123523 (2008).
[16] H. Oyaizu, M. Lima and W. Hu, Phys. Rev. D, **78**, 123524 (2008).
[17] F. Schmidt, M. Lima, H. Oyaizu and W. Hu, Phys. Rev. D, **79**, 083518 (2009).
[18] F. Schmidt, Phys. Rev. D, **80**, 043001 (2009).
[19] A. Knebe, A. G. Green and J. Binney, Mon. Not. R. As-

tron. Soc., **325**, 845 (2001).
[20] B. Li and H. Zhao, Phys. Rev. D**80**, 044027 (2009).
[21] B. Li and H. Zhao, Phys. Rev. D **81**, 104047 (2010).
[22] G. Zhao, B. Li and K. Koyama, Phys. Rev. D**83**, 044007 (2011).
[23] B. Li and J. D. Barrow, Phys. Rev. D**83**, 024007 (2011).
[24] B. Li, D. F. Mota and J. D. Barrow, Astrophys. J., **728**, 108 (2011).
[25] B. Li, D. F. Mota and J. D. Barrow, Astrophys. J., **728**, 109 (2011).
[26] P. Brax, C. van de Bruck, A. -C. Davis, B. Li and D. J. Shaw, Phys, Rev. D**83**, 104026 (2011).
[27] A. -C. Davis, B. Li, D. F. Mota and H. A. Winther, Astrophys. J., submitted; arXiv:1008.3082 [astro-ph.CO].
[28] R. Teyssier, Astron. Astrophys. **385** (2002) 337-364.
[29] W. Hu and I. Sawicki, Phys. Rev. D, **76**, 064004 (2007).
[30] H. Martel and P. R. Shapiro, Mon. Not. R. Astron. Soc., **297**, 467 (1998).
[31] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, published by the Taylor & Francis Group (1988).
[32] R. E. Smith *et al.* [The Virgo Consortium Collaboration], Mon. Not. Roy. Astron. Soc., **341**, 1311 (2003).
[33] T. Guillet and R. Teyssier, J. .Comp. Phys., **230**, 4756 (2011).
[34] S. Prunet, C. Pichon, D. Aubert, D. Pogosyan, R. Teyssier and S. Gottloeber, arXiv:0804.3536 [astro-ph].
[35] S. Colombi, A. H. Jaffe, D. Novikov and C. Pichon, arXiv:0811.0313 [astro-ph].