Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **434,** 2094–2120 (2013)
Advance Access publication 2013 July 23

doi:10.1093/mnras/stt1154

A new way of setting the phases for cosmological multiscale Gaussian initial conditions

Adrian Jenkins★

Institute for Computational Cosmology, Department of Physics, University of Durham, South Road, Durham DH1 3LE, UK

Accepted 2013 June 21. Received 2013 June 20; in original form 2013 April 12

ABSTRACT

We describe how to define an extremely large discrete realization of a Gaussian white noise field that has a hierarchical structure and the property that the value of any part of the field can be computed quickly. Tiny subregions of such a field can be used to set the phase information for Gaussian initial conditions for individual cosmological simulations of structure formation. This approach has several attractive features: (i) the hierarchical structure based on an octree is particularly well suited for generating follow-up resimulation or zoom initial conditions; (ii) the phases are defined for all relevant physical scales in advance so that resimulation initial conditions are, by construction, consistent both with their parent simulation and with each other; (iii) the field can easily be made public by releasing a code to compute it – once public, phase information can be shared or published by specifying a spatial location within the realization. In this paper, we describe the principles behind creating such realizations. We define an example called Panphasia and in a companion paper by Jenkins and Booth (2013) make public a code to compute it. With 50 octree levels Panphasia spans a factor of more than 10^{15} in linear scale – a range that significantly exceeds the ratio of the current Hubble radius to the putative cold dark matter free-streaming scale. We show how to modify a code used for making cosmological and resimulation initial conditions so that it can take the phase information from Panphasia and, using this code, we demonstrate that it is possible to make good quality resimulation initial conditions. We define a convention for publishing phase information from Panphasia and publish the initial phases for several of the Virgo Consortium's most recent cosmological simulations including the 303 billion particle MXXL simulation. Finally, for reference, we give the locations and properties of several dark matter haloes that can be resimulated within these volumes.

Key words: methods: numerical – cosmology: theory.

1 INTRODUCTION

Computer simulations are the main tool for exploring the predictions of models of cosmological structure formation in the strongly non-linear regime. According to the currently favoured model, Λ cold dark matter (ΛCDM), the structure we see today in the Universe is seeded from small adiabatic Gaussian fluctuations believed to have originated in a very early inflationary phase in the Universe's history. Non-linear structure formation is simulated in ΛCDM by making initial conditions at an epoch before there is significant non-linearity, followed by integrating the equations of motion forward in time using for example an *N*-body code. The creation of the initial conditions requires a method to produce realizations of Gaussian random fields. The techniques to make such realizations have advanced in tandem with simulation methods over the last three decades.

Simulation work exploring CDM in the 1980s focused on *N*-body modelling of representative volumes of the Universe to study the large-scale structure in model universes made of collisionless dark matter. The Gaussian initial conditions for these simulations were made by using Fourier methods applied to a single cubic mesh (Efstathiou et al. 1985). These simulations modelled periodic cubic domains of space sampled with uniform mass particles throughout the volume.

The continuing success of the CDM model led in the mid-1990s to the need to set up more complex simulations to explore the model deeper into the non-linear regime including the study of the internal structure of dark matter haloes. Because haloes could not be adequately resolved at reasonable computational cost in cosmological simulations, new techniques had to be developed to 'resimulate' haloes more cheaply. The initial conditions for the resimulations needed both to faithfully reproduce the target haloes, selected from a representative cosmological volume, and to include extra small-scale power that could not be resolved in the parent simulation. The

★E-mail: a.r.jenkins@durham.ac.uk

computational cost was minimized by coarse sampling the majority of the simulation volume with massive particles whose function was to provide the correct tidal environment for the halo being formed. New codes were developed to make these resimulation initial conditions: to build multimass particle loads and to make multiscale realizations of Gaussian fields.

These first codes, such as the one described in Navarro, Frenk & White (1995), used Fourier methods to generate the multiscale Gaussian realizations. In this example, the field is built in a two-step process. First, the Fourier modes present in the original cosmological simulation are regenerated – guaranteeing that the target halo is reproduced. Secondly, extra small-scale power is generated. The total displacement field for the particles is the sum of the contributions from both fields. The reason for needing two steps is that it is usually impractical to use a larger Fourier transform for making the resimulation initial conditions than was used to generate the initial conditions of the parent simulation. To get round this practical limitation, the grid for the small-scale power is made physically smaller and placed around just the region that forms into the halo and its immediate surroundings. Typically, this second grid is an order of magnitude smaller than the original grid. Over the next decade, this method was extended to allow the placing of further nested grids. This made it possible to reach very high numerical resolution in cosmological simulations. Using seven concentric grids, Gao et al. (2005) achieved sub-solar particle mass resolution for a small patch within 479 h^{-1} Mpc on a side periodic volume.

While the Fourier method has proved to be a popular method for making resimulations for nearly two decades, it has, since the late 1990s, coexisted with an alternative approach for making multiscale Gaussian initial conditions suggested by Salmon (1996). Salmon pointed out that it is easy to make a discrete realization of a real-space multiscale Gaussian white noise field with the aid of any standard pseudo-random generator. This is because the values of a Gaussian white noise field at different points are independent and so can be set up sequentially. Once such a white noise field has been generated, it can be transformed by convolving it with an appropriate filter to produce a Gaussian realization with any other power spectrum. This idea gave birth to what we will call the real-space white noise field method for making multiscale Gaussian fields – it was adopted by Pen (1997) and Bertschinger (2001). Bertschinger released a public code called GRAFIC2 based on this idea that can generate multiscale Gaussian fields. This code has been used by others to generate resimulation initial conditions. A parallel version of GRAFIC2 has been developed by Stadel et al. (2009), who used it to set up the GHALO series of halo simulations. Most recently Hahn & Abel (2011) have refined the real-space white noise field method by applying a new algorithm that uses an adaptive convolution to improve the accuracy of the numerical convolution by two orders of magnitude when compared to the Fourier methods deployed in the GRAFIC2 code.

To make resimulation initial conditions using a real-space Gaussian white noise field, it is necessary to be able to refine the white noise field in any region of interest to allow finer spatial scales to be resolved. This refined patch of the white noise field must remain consistent with the unrefined white noise it is replacing to ensure that the same structures are reproduced. Ideally, this process of refining would operate as if some predefined underlying Gaussian white noise field were simply being revealed in more and more detail with increasing levels of refinement rather than simply being invented to order. This idea is only partially achieved in the current approaches to refinement described in the literature.

Typically the process of refining starts with a discretized Gaussian white noise field that is specified on a cubic grid of coarse cells. Each coarse cell is labelled by a single field value that is associated with the cell centre. Each field value is picked at random from a Gaussian distribution with zero mean and a variance that is proportional to the inverse of the cell volume. A refined version of the field is produced for a grid of finer cells. These fine cells are generated by splitting the coarse cells in a patch of interest into smaller equal-sized cubic cells. In Pen (1997) and Bertschinger (2001), the Gaussian values over the fine grid are chosen freely in the same way as the coarse cells, but subject to the constraint that the sum of the values over the fine cells within each coarse cell equals the value associated with that coarse cell. While elegant, this approach does not guarantee that the corresponding Fourier modes for the patch as a whole taken in isolation are the same for the coarse and fine versions. A more complex approach also using linear constraints has been developed by Hahn & Abel (2011) which forces the corresponding Fourier modes in the coarse and fine overlapping region to agree. While the latter approach guarantees that the large-scale power on the patch is precisely reproduced in the refinement, neither method succeeds in defining a truly objective white noise field – that is one that is independent of the details of how the refinements are laid down. This lack of objectivity applies equally to the Fourier method of making initial conditions as the precise placement, and phases used for the additional grids are arbitrary.

The goal of this paper is to develop a practical way to create a realization of an objective Gaussian white noise field. There are two major advantages for having such a Gaussian white noise field: (i) it guarantees that sets of initial conditions made for the same region with different numerical resolutions are as consistent as they can be – so that the process of performing successive resimulations becomes one closer to discovery – as the structures on all physical scales are predefined; (ii) if the Gaussian white noise field is made public, then it becomes easy to share or publish the phase information. All that is needed is to give the precise location of the phase information within the realization as a whole. The routine publishing of the phase information has the potential to enrich the literature by making it easier for others to check, reproduce or build upon published simulation work.

The reason why Gaussian white noise fields are convenient to work with numerically is one particular property. This is the fact that the expansion coefficients of a Gaussian random white noise field with respect to any orthogonal basis function expansion are necessarily independent Gaussian variables. This property means that a realization of Gaussian white noise fields can be conveniently created by using a pseudo-random number generator to set the values of the expansion coefficients.

The fact that this property is true for any orthogonal basis function set suggests that it might be possible to find a set of orthogonal basis functions that are particularly well suited for making cosmological initial conditions. Neither Fourier modes nor sets of independent values arranged on a real-space grid are obviously optimal for the task of generating cosmological multiscale Gaussian fields.

The ability to resimulate any region of a simulation to any desired numerical resolution requires that it must be possible to successively refine the Gaussian white noise field at an arbitrary location. This requirement suggests looking for an orthogonal basis set with a hierarchical structure. An octree, which is the set of cells formed by dividing a cube into eight subcubes and continuing this operation recursively on the subcubes, would seem the simplest and most convenient geometrical structure to adopt. Using this structure we can define ‘orthogonal octree basis functions’ to be functions that

are localized to particular octree cells being zero everywhere else. Clearly defined in this way, octree basis functions in octree cells that do not overlap are trivially orthogonal. For octree cells that do overlap, the requirement of orthogonality is non-trivial and limits the possible functional forms for the octree basis functions as we show later. The high-symmetry and self-similar nature of an octree means in practice that a relatively small set of functional forms are needed to describe the infinite set of octree basis functions needed to populate an octree to unlimited depth.

We can define the phase information for a periodic cosmological simulation in the following way. We identify the cubic volume of the simulation with a corresponding cubic subvolume within the octree. This region in the octree can be made of a single octree cell or a group of cells. We can then use the values of the Gaussian white noise field in this chosen subvolume of the octree to define the phases for the cosmological simulation to any desired resolution. Because a Gaussian white noise field within a region is completely independent of the field outside of that region, we can effectively cut out conveniently sized cubic blocks from a much large Gaussian white noise field and use these independent blocks to define the phases for particular cosmological volumes.

The octree functions form a discrete four-dimensional space – three of these dimensions span physical space in the form of a cubic grid consisting of eight to some integer power cells, while the fourth dimension spans the allowed side lengths of the octree cells which are the side length of the root cell divided by two to some integer power. A realization of a Gaussian white noise field can be made using these octree basis functions by first establishing a 1D to 4D mapping between a pseudo-random Gaussian number sequence and the space of octree functions. Each pseudo-random number is taken as the expansion coefficient of the white noise field for a particular octree basis function. Similar mapping strategies, although more commonly from 1D to 3D, are used in all methods used to make Gaussian cosmological initial conditions.

To exploit the full potential of this 1D to 4D mapping so that it is possible to refine the white noise field at any location to any depth, it must be possible to access all of the relevant expansion coefficients at reasonable computational cost. This requires choosing a pseudo-random number generator that allows large jumps through the linear pseudo-random sequence to be made cheaply. Fortunately there are classes of pseudo-random number generators with this property, and amongst these there are well-tested generators in common use.

Given that it is possible to access any expansion coefficient relatively easily, it becomes possible by assigning the entire period of the pseudo-random generator to the octree to create a realization of a Gaussian white noise field with a truly enormous dynamic range. The typical periods of generators commonly in use are so large that the resulting white noise field is far larger than needed for any one simulation, or indeed for all simulations that have ever been run (at least on this planet!). For most generators, the period is big enough to define a white noise field that can resolve scales below the putative CDM free-streaming scale (Hofmann, Schwarz & Stöcker 2001) everywhere within a volume that greatly exceeds the current Hubble volume.

The rest of this paper is a detailed elaboration of the ideas outlined in this introduction leading to the construction of a particular realization, called Panphasia, which is designed for the purpose of making accurate cosmological and resimulation initial conditions. We make this field public in a companion paper (Jenkins & Booth 2013). The outline of the rest of the paper is as follows: in Section 2, we will give the mathematical background to the properties of Gaussian fields needed later. In Section 3, we give a general description of how to construct the octree orthogonal basis functions and outline their properties and choose the most suitable set for making simulation initial conditions. We save the nitty-gritty and more tedious details of the practical implementation to Appendix A. In Section 4, we introduce the pseudo-random number generator and its properties, but leave the details of the precise mapping of the sequence to the octree to Appendix B. In Section 5, we show how to add Panphasia to the IC_2LPT_GEN initial conditions code first described in Jenkins (2010). In Section 6, we generate and test cosmological and resimulation initial conditions and show that it is possible to obtain good results using Panphasia. In Section 7, we define a convention for publishing phases and give the phases for several of the most recent Virgo Consortium volumes together with the locations of a few haloes within these volumes. In Section 8, we give an overview of the code to compute Panphasia. Section 9 gives the summary. Finally in Appendix C we give the formal definition of Panphasia.

2 MATHEMATICAL BACKGROUND

2.1 Orthogonal basis function expansions of Gaussian white noise fields

In the ΛCDM model, the primordial density fluctuations are a homogeneous and isotropic Gaussian field. Taking the spatial curvature to be negligible, a ΛCDM universe can be modelled as a finite cube of side length L, with periodic boundary conditions. For such a cube, we can describe the density fluctuations in terms of the matter overdensity, $\delta(\boldsymbol{x})$, where $\boldsymbol{x}$ is the position, as a sum over Fourier modes:

$$\delta(\boldsymbol{x}) = \sum_{\boldsymbol{k}} \delta(\boldsymbol{k}) \exp\left[\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}\right]. \tag{1}$$

The periodic boundary conditions require the wavevector, $\boldsymbol{k}$, to take discrete values with Cartesian components $(k_x, k_y, k_z) = (2\pi/L)(l_x, l_y, l_z)$, where l_x, l_y, l_z are integers.

By definition, a Gaussian field is one where the amplitudes of the Fourier modes, $\delta(\boldsymbol{k})$, are independent, with the real and imaginary parts of each mode drawn from the same Gaussian distribution. The statistical properties of a Gaussian random field are completely determined by its power spectrum, which is defined by

$$P(\boldsymbol{k}) = \langle|\delta(\boldsymbol{k})|^2\rangle/L^3, \tag{2}$$

where the angular brackets signify an ensemble average. For a real Gaussian field with zero mean overdensity, the Fourier mode amplitudes obey the constraints $\delta(\boldsymbol{k} = (0,0,0)) = 0$ and $\delta(\boldsymbol{k}) = \delta^*(-\boldsymbol{k})$, where δ^* is the complex conjugate of δ.

By definition, a Gaussian white noise field has a constant power spectrum. For convenience, we will take $\langle|\delta(\boldsymbol{k})|^2\rangle = 1$ for all Gaussian white noise fields in this paper. There are two properties of Gaussian white noise fields that are particularly relevant for this paper.

First, a white noise field has power at all wavenumbers which means that it can always be transformed into another Gaussian field with any desired power spectrum by convolving it with a suitable kernel function. As is well known, the operation of a spatial convolution corresponds in Fourier space to a simple scaling of the amplitude of each of the Fourier modes. By choosing a scaling factor with a modulus of $\sqrt{P(\boldsymbol{k})}$, the power spectrum of the convolved white noise field becomes $P(\boldsymbol{k})$. The scaling can include an arbitrary phase factor, but as the goal of this paper is to use a white noise field to define the phase, we need to insist that the kernel function

must be real and non-negative in Fourier space. With this choice, the phase information is purely contained in the white noise field itself.

The second property, which we derive here, is the fact that the basis function coefficients of any orthogonal basis function expansion of a Gaussian white noise field are necessarily independent Gaussian variables. Our definition of a Gaussian field asserted that the expansion coefficients are independent Gaussians for a plane wave expansion, and the plane waves are an example of an orthogonal basis function set. We now show that given that this is true for a plane wave expansion of a Gaussian white noise field, it is also true for all other sets of orthogonal basis functions.

To show this, we first consider a vector $\boldsymbol{V}$ with N components, v_i, $i = 1, N$, which are Gaussian independent random variables with $\langle v_i \rangle = 0$ and $\langle v_i v_j \rangle = \delta_{ij}$. When we say the variables are independent, we mean that the joint probability distribution of the N variables is

$$\mathrm{Prob}(v_1, v_2, \ldots, v_n) = \frac{1}{(2\pi)^{N/2}} \exp\left[-\frac{1}{2}\boldsymbol{V}^T\boldsymbol{V}\right], \tag{3}$$

which is just the product of the N individual Gaussian probability distributions.

Consider now an alternative vector $\boldsymbol{W}$, with N components w_i, $i = 1, N$ which are linearly related to the v_i by

$$\boldsymbol{W} = \mathbf{R}\boldsymbol{V}, \tag{4}$$

where the matrix $\mathbf{R}$ is any orthonormal matrix, so that $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}$.

As the magnitude of the Jacobian for a linear transformation between the v_i and w_i variables is just the magnitude of the determinant of $\mathbf{R}$, which is unity for an orthonormal matrix, we can simply transform equation (4) to give the joint probability distribution of the w_i:

$$\mathrm{Prob}(w_1, w_2, \ldots, w_n) = \frac{1}{(2\pi)^{N/2}} \exp\left[-\frac{1}{2}\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}\right], \tag{5}$$

which shows that the w_i are independent Gaussian variables too.

To apply this result to Gaussian white noise fields themselves, we need to go from using finite vectors to using infinite vectors to represent functions in Hilbert space, in the manner made familiar by quantum mechanics.

Let the $f_i(\boldsymbol{x})$ be an infinite set of real functions defined over the periodic volume which obey these orthogonality and normalization relations:

$$\int f_i(\boldsymbol{x}) f_j(\boldsymbol{x}) \mathrm{d}^3\boldsymbol{x} = \delta_{ij}, \tag{6}$$

where the indices i, j label the functions, and the integral is over the volume. Using Parseval's relation the corresponding normalization/orthogonality relations in Fourier space are

$$\sum_{\boldsymbol{k}} \tilde{f}_i(\boldsymbol{k}) \tilde{f}_j(\boldsymbol{k}) = \delta_{ij}, \tag{7}$$

where the function $\tilde{f}_i$ is the Fourier transform of f_i. If we express a Gaussian white noise field as an expansion in the functions f_i, with expansion coefficients C_i, and assume that these functions form a complete set, then we can express the overdensity field as a sum over these basis functions

$$\delta(\boldsymbol{x}) = \sum_j C_j f_j(\boldsymbol{x}), \tag{8}$$

where the sum is over the whole set of functions.

Formally this sum is ill defined for a white noise field, but nonetheless this expression can be employed inside of integrals in a well-defined way. An expression for the C_i can be obtained by multiplying both sides by f_i and integrating over all space, applying Parseval's relation to the l.h.s. and equation (6) to the r.h.s. to give

$$C_i = \sum_{\boldsymbol{k}} \tilde{f}_i(\boldsymbol{k}) \delta(\boldsymbol{k}), \tag{9}$$

where the sum is over all wavenumbers. We can recognize this equation as being an orthonormal transformation, analogous to equation (4) between the plane wave set of expansion coefficients, $\delta(\boldsymbol{k})$, and the C_i expansion coefficients of the f_i orthogonal basis function set. Thus, the C_i must be independent Gaussian variables.

Having shown this, we can define a Gaussian white noise field in terms of the C_i coefficients. By using a pseudo-random generator to assign values to the C_i, we can create a realization of a Gaussian white noise field in terms of the f_i functions.

To date only two sets of orthogonal basis functions have been used for making cosmological simulation initial conditions. These are the plane waves and real-space grids of delta functions.

Plane waves are attractive because it is possible to refine a Gaussian white noise field simply by adding higher wavenumber modes to the modes already present. Their disadvantage is that this approach is very expensive computationally. This is because the plane waves are not localized in real space. This means to refine a white noise field in one region, it is necessary to provide the information to refine the field everywhere to the same degree. This high cost has been avoided by for example Navarro et al. (1995) by placing a smaller Fourier grid around the region of interest so that the newly added modes only contribute to a small part of the simulation volume. This approach is necessarily approximate as the added modes are not truly orthogonal to the original plane waves.

By contrast, arrays of delta functions are perfectly localized, which makes them very efficient from a computational point of view. However, their disadvantage is that they are not able to resolve scales smaller than the distance between adjacent delta functions and cannot simply be added to generate a finer version of the field. The approach in the literature (Pen 1997; Bertschinger 2001; Hahn & Abel 2011) to get round this limitation has been to simply replace one set of delta functions with an 'equivalent' finer set. This equivalence is achieved by using a set of linear constraints to force the phase information to be as similar as possible between the original and its replacement. This approach is also approximate.

However, it is possible to find orthogonal basis functions that have a strong degree of locality like the real-space delta functions and enable a Gaussian white field to be refined just by adding more components as with the plane waves. In this paper, we develop a set of orthogonal octree basis functions that have these properties. The octree basis functions are spatially extended, but each is localized to a single cell of the octree. Using them a Gaussian white noise field can be refined at relatively low computational cost by just adding new basis functions from deeper in the octree to the existing field in the region of interest only.

Before discussing the nature of the basis functions, we first define some terminology to describe the octree structure we need.

2.2 Notation to describe an octree

We identify the root cell of the octree with the whole periodic cubic spatial domain of length L. We define a set of Cartesian coordinates, (x_1, x_2, x_3), aligned with the three orthogonal edges of the root cell and place the corner at (0, 0, 0). The coordinates have an allowed range $0 \leq x_i < L$, $i = 1, 2, 3$. We define the root cell to be at level 0 of the octree.

At level l of the octree, there are 8^l cubes each of side length

$$\Delta_l = \frac{L}{2^l}. \tag{10}$$

We will label these cells using integer Cartesian coordinates, j_1, j_2, j_3 where $0 \leq j_i < 2^l$, $i = 1, 2, 3$. The centre of a cell (j_1, j_2, j_3) at level l, $\boldsymbol{x}_c$, has coordinates

$$\boldsymbol{x}_c(l, j_1, j_2, j_3) = (j_1 + 1/2, j_2 + 1/2, j_3 + 1/2)\Delta_l. \tag{11}$$

Each octree cell has eight child cells, and we will call this cell the parent cell with respect to any of its child cells.

3 BUILDING THE ORTHOGONAL OCTREE BASIS FUNCTIONS

The primary objective of this paper is to use a new orthogonal basis function set based on an octree structure to construct a realization of a Gaussian white noise field. By definition, these octree basis functions are localized to particular octree cells and mutually orthogonal. We require that the infinite set of octree basis functions can be described by a finite set of functional forms that are common to all levels of the tree, save for a normalization constant that is allowed to depend on the level of a cell in the octree.

It proves convenient numerically to define the octree basis functions themselves in terms of a smaller set of more primitive building block functions. The octree basis functions can all be built from different combinations of these building blocks. These building block functions are similarly localized to particular octree cells and, as described later, are each separable into the product of three one-dimensional functions of the Cartesian coordinates. An approximation to a Gaussian white noise field can be constructed using these blocks by placing one or more of these functions into each octree cell at level l of the octree so that they collectively tessellate the entire volume. Each function is given an individual weight, which is determined by the Gaussian white noise field. For convenience, the functional forms of these building blocks are chosen so that they form an orthogonal basis set when occupying the octree cells at a single level of the octree. We can use these blocks to approximate a Gaussian white noise field by making a basis function expansion of the field with these blocks. The expansion coefficients of these blocks will be independent Gaussian variables as the blocks are orthogonal.

This representation is an approximation to a white noise field because no finite set of building block functions can form a complete basis set. For cosmological initial conditions, we require that the density fluctuations be accurate from large scales down to some minimum lengthscale determined by numerical reasons such as by the particle Nyquist frequency or the gravitational softening length. For this basis function expansion of a Gaussian white noise field to be a useful approximation for our purposes, we require that the power spectrum tends to unity in the limit $k \to 0$ so that the power on large scales is accurately represented. If this condition is met, then it becomes possible to generate an accurate approximation of a Gaussian white noise field for all wavevectors with a modulus below some given value provided the building block functions are placed at a sufficient depth in the octree. It is not difficult to see that a building block function that is just a constant within an octree cell and zero everywhere else is one possibility. As we will see later, we can add additional building block functions to improve the rate of convergence to a white noise spectrum in the limit $k \to 0$.

Another practical requirement on the possible set of building block functions is that it must be possible to construct each of these functions when placed in an octree cell at level l, out of a superposition of the same set of building blocks placed at level $l + 1$ in the eight child cells. If this is true and we assume that the octree functions at level l can be built from the building block functions placed at some deeper level, then it follows that all of the octree basis functions from level $l - 1$ to the root cell can similarly be exactly represented by these building block functions at this one level. In other words, it is possible to project the four-dimensional space of octree basis functions, above any given depth, on to a three-dimensional grid and represent it exactly using the building block functions placed in these grid cells with appropriate weightings.

In fact, the octree basis functions are to a large extent implicitly defined by the choice of these building block functions. To see this imagine taking a given realization of a Gaussian white noise field and approximating it in two different ways as a basis function expansion in a given set of building blocks at either level l or at level $l + 1$. The expansion at level $l + 1$ will contain all the information present in level l expansion, but not vice versa. The additional information about the white noise field at level $l + 1$ is just that introduced by the addition of a single layer of octree basis functions. It follows from this that the octree functions, which occupy whole octree cells at level l, must each be built of eight blocks with each child cell containing some superposition of the building block functions and that they are orthogonal to the level l building blocks. As the expansion at level $l + 1$ has the number of expansion coefficients eight times that at level l, it follows that the number of independent octree functions must be seven times the number of building block functions. By a similar argument, we can deduce that the ensemble average power spectrum of a layer of octree basis functions placed at level l of the octree is simply given by the difference between the ensemble average power spectrum of a white noise field expanded using the building blocks at levels l and $l + 1$.

While it may not be immediately obvious, it is not difficult to see that it is possible to create sets of building blocks with the required properties starting from Legendre polynomials.

3.1 Using Legendre functions to build the octree basis functions

The Legendre polynomials, by definition, are localized in a finite interval and obey the orthogonality relation

$$\int_{-1}^{1} P_l(x) P_m(x) \mathrm{d}x = \frac{2}{2l+1}\delta_{lm}. \tag{12}$$

The lowest order Legendre polynomials are $P_0(x) = 1$, $P_1(x) = x$ and $P_2(x) = (3x^2 - 1)/2$.

We can define three-dimensional 'Legendre block' functions as products of Legendre polynomials in all three Cartesian coordinates within a unit length cubic cell and zero outside as follows:

$$p_{j_1 j_2 j_3}(\boldsymbol{x}) = \begin{cases} \prod_{i=1}^{3} (2j_i + 1)^{1/2} P_{j_i}(2x_i) & \text{if } -\frac{1}{2} \leq x_i < \frac{1}{2}; \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

The Legendre blocks, when placed within octree cells at a given level in an octree, are orthogonal and obey the following normalizations and orthogonality relations when integrated over all space:

$$\int p_{i_1 i_2 i_3}(\boldsymbol{x}) p_{j_1 j_2 j_3}(\boldsymbol{x}) \mathrm{d}^3\boldsymbol{x} = \delta_{i_1 j_1}\delta_{i_2 j_2}\delta_{i_3 j_3}. \tag{14}$$

All of the Legendre blocks meet the requirement for the building blocks to be localized and orthogonal. The requirement that each block can be built of a superposition of blocks deeper within the octree does place some constraints on the suitable sets of blocks. We will refer to a set of blocks with this property as being self-representing.

It is not difficult to see that it is possible to define sets of blocks that are self-representing by taking all possible blocks built from combinations Legendre polynomials up to some given order. We will call the S_1 set the set that was made from P_0 alone, S_8 that was made from the eight combinations of the P_0 and P_1 polynomials and S_{27} that was made from the 27 combinations of the P_0, P_1 and P_2 polynomials. We will refer to the S_1, S_8 and occasionally the S_{27} sets of Legendre blocks throughout the rest of the paper. We will use these labels also to refer to the octree basis function themselves, so when we refer to for example the S_8 octree basis functions, we mean those that are built from the S_8 set of Legendre block functions.

Having defined some potential sets of Legendre blocks for building a Gaussian white noise field, we need some way to judge their relative merits. While all Legendre blocks contribute equally to the total variance of the white noise field (in an ensemble averaged sense), they differ in their relative contributions to the power spectrum as a function of wavenumber. To examine this further, we need to look at the Fourier representations of the Legendre blocks.

The Fourier transform of the Legendre polynomials in the $[-1, 1]$ interval is the spherical Bessel functions. The lowest order spherical Bessel functions are $j_0(k) = \sin(k)/k$, $j_1(k) = (\sin(k) - k\cos(k))/k^2$. As $k \rightarrow 0$ so $j_n(k) \rightarrow k^n/(2n+1)!!$. We define the Fourier transform of the Legendre blocks as follows:

$$\int_{L^3} p_{j_1 j_2 j_3}(\boldsymbol{x}) \exp[\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}]\mathrm{d}^3\boldsymbol{x} = i^n j_{j_1 j_2 j_3}(\boldsymbol{k}), \tag{15}$$

where the functions $j_{j_1 j_2 j_3}$ are related to the spherical Bessel functions:

$$j_{j_1 j_2 j_3}(\boldsymbol{k}) = \prod_{i=1,3} (2j_i + 1)^{1/2} j_{j_i}\left(\frac{k_i}{2}\right). \tag{16}$$

The spherical Bessel functions obey an identity for all k:

$$\sum_{l=0}^{\infty} (2l+1) j_l^2(k) = 1. \tag{17}$$

Similarly the functions $j_{j_1 j_2 j_3}$ obey

$$\sum_{j_1=0}^{\infty}\sum_{j_2=0}^{\infty}\sum_{j_3=0}^{\infty} j^2_{j_1 j_2 j_3}(\boldsymbol{k}) = 1. \tag{18}$$

We will need this identity later to establish the completeness of the octree basis functions.

It is possible to have two self-representing sets of Legendre blocks, S_M and S_N with $M < N$, where the former set of blocks is a subset of the latter. For example, S_8 is a subset of S_{27}. In such cases, given a realization of a Gaussian white noise field that has been constructed using the S_N octree basis functions, it is possible to obtain an equivalent representation of exactly the same field but built from the S_M octree basis functions by simply ignoring those Legendre blocks that are not part of S_M. While this might seem paradoxical, this equivalence is only true for expansions made with complete sets of octree basis functions which are infinite in number. For expansions made with a finite set of octree basis functions, some choices are better than others as judged for the purposes of making initial conditions. We will show this later by comparing simulations of a particular dark matter halo at redshift zero run from initial conditions made using either S_1 or S_8 octree basis functions.

The reverse procedure of starting with a field based on the set S_M and wanting to create an equivalent field but using the superset S_N is non-trivial. The coefficients of the blocks that are part of S_N but not S_M are implicitly determined by an infinite number of coefficients belonging to the S_M blocks at deeper levels of the tree. For this reason, it is better to be somewhat conservative in the initial choice of sets of Legendre block functions and to try and take as large a set as might possibly be needed. Taking too large a set however risks making the generation of the field needlessly slow. For this paper, we will evaluate just the S_1 and S_8 octree basis functions. As will be shown later, the former does not perform very well and the latter performs well enough that there is no compelling reason to look at more elaborate choices.

3.2 Properties of the octree basis functions

We can write the ensemble power spectrum of a basis function expansion of a Gaussian white noise field using the set S_N of Legendre blocks at level l of the octree as

$$\left\langle P_l^N(\boldsymbol{k})\right\rangle = \sum_{S_N} j^2_{j_1 j_2 k_3}(\boldsymbol{k}\Delta_l), \tag{19}$$

where Δ_l is the size of the octree cells, defined in equation (10).

The j_{000} function, present in all sets, tends to unity as $k \rightarrow 0$, while the sum over the whole set of Legendre block functions, given by equation (18), is unity for all wavenumbers. We can therefore see that the power spectrum approaches unity from below as $\boldsymbol{k}\Delta_l \rightarrow 0$. Similarly, the power spectrum tends to unity from below for any given $\boldsymbol{k}$ as $\Delta_l \rightarrow 0$. This demonstrates that the set of octree functions is complete: the Fourier mode set is complete and the amplitude of any Fourier mode can be reproduced by the octree basis functions in the limit of infinite tree depth.

For practical purposes the tree depth is finite, and there are significant differences between how well different sets of octree basis functions approximate the large-scale modes of a white noise field. As mentioned at the beginning of this section, we can determine the ensemble power spectrum of a single level of octree basis functions by taking the difference between the ensemble average power spectrum of the building blocks placed at two adjacent levels of the octree:

$$\left\langle P^N_{l-1,\mathrm{oct}}(\boldsymbol{k})\right\rangle = \sum_{S_N}\left(j^2_{j_1 j_2 j_3}(\boldsymbol{k}\Delta_l) - j^2_{j_1 j_2 j_3}(2\boldsymbol{k}\Delta_l)\right). \tag{20}$$

Note we have implicitly associated a level $l-1$ for the octree basis functions that are made from eight level l Legendre block functions.

Fig. 1 shows the ensemble average octree power spectrum for a single octree layer for the S_1 and S_8 sets of Legendre blocks. The power spectrum shown is an average over cubic shells in k-space. The power in S_8 is more sharply peaked with the logarithmic slope at low $k\Delta$ of 4, compared to 2 for S_1. At high $k\Delta$, the logarithmic slopes of both are -4, but as the S_8 set contains eight times as many basis functions as S_1, it has eight times more power and therefore has a significantly higher amplitude at higher k. The three-dimensional power spectrum of the octree functions has cubic symmetry and is therefore anisotropic. We will leave the practical issue of how to restore isotropy on all scales to later in the paper when we describe how to make initial conditions.

The deviation of the power spectrum from unity at low k for the S_8 set scales as k^4 which is significantly better than the k^2 scaling of the S_1 set. This makes the S_8 set significantly better at approximating

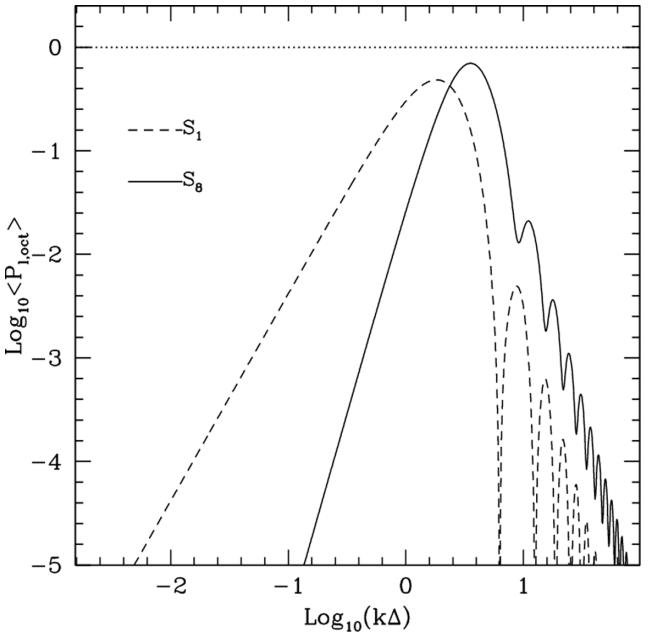


Figure 1. The ensemble average power spectrum of the octree basis functions, given by equation (20), at a single level of the octree for two different choices of Legendre blocks. The power spectrum given is the average over a cubic shell centred on the origin and labelled by the maximum Cartesian coordinate. The horizontal dotted line shows the amplitude of the white noise field. Both S_1 and S_8 have a logarithmic slope of -4 at large values of $k\Delta$. The logarithmic slopes at small values are 2 and 4, respectively.

the large-scale power. By this measure the S_{27} set would be even better with its k^6 scaling, but there are disadvantages in using large block sets.

Using a larger set of Legendre blocks incurs a greater computational expense when evaluating the field. This cost is made up of two components: the extra pseudo-random numbers that need to be computed, which scale linearly with the number of Legendre blocks, and the time to compute the relevant Legendre coefficients from the octree basis functions, which involves linear algebra with a matrix whose size scales as the square of the number of Legendre blocks. For the S_8 set, these two elements take similar amounts of cpu time. We would expect that S_{27} would be about a factor of roughly 10 more expensive than S_8 to evaluate per octree cell. For practical reasons, discussed later, it would also be necessary to evaluate the S_{27} field more times to avoid the code being any more memory intensive and that could make it 30 times more expensive. This view is informed by the performance of the code which we make public in Jenkins & Booth (2013). If a significantly faster code could be developed to compute the field, then the practical argument against using the S_{27} set would be weakened.

In the next subsection, we explicitly define sets of the S_1 and S_8 octree basis functions.

3.3 Functional forms for the S_1 and S_8 octree functions

While the octree basis functions are three-dimensional functions, they can be factorized into products of three one-dimensional functions of each of the Cartesian coordinates. For S_1 these one-dimensional functions are built from the P_0 Legendre polynomial, which is just a constant, while for S_8 the P_0 and P_1 Legendre polynomials are required. For S_1 we define two one-dimensional functions:

$$D_0(u) = \begin{cases} 1 & \text{if } -1 \le u < 1; \\ 0 & \text{otherwise,} \end{cases} \tag{21}$$

$$D_1(u) = \begin{cases} 1 & \text{if } 0 \le u < 1; \\ -1 & \text{if } -1 < u < 0; \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Using these functions we can generate eight three-dimensional functions, F_{ijk}, occupying an octree cell at level l as follows:

$$F^l_{ijk}(\boldsymbol{x}) = \frac{1}{\Delta_l^{3/2}} D_i\left(\frac{2x_1}{\Delta_l}\right) D_j\left(\frac{2x_2}{\Delta_l}\right) D_k\left(\frac{2x_3}{\Delta_l}\right), \tag{23}$$

where i, j, k are the integers either zero or one, x_1, x_2, x_3 are the Cartesian components of $\boldsymbol{x}$ and it is assumed that the origin is the centre of the octree cell at level l. We can consider all of these functions of being built from eight p_{000} Legendre blocks placed, with appropriate weights, in the child cells at level $l+1$.

The function F^l_{000} is just a constant and corresponds to a p_{000} Legendre block at level l. The seven other functions are the octree basis functions themselves. Note that each octree function has at least one discontinuity in value. Given that the functions D_0 and D_1 are, respectively, symmetric and antisymmetric about the origin, it follows that all eight functions are mutually orthogonal:

$$\int F^l_{ijk}(\boldsymbol{x}) F^l_{lmn}(\boldsymbol{x}) \mathrm{d}^3\boldsymbol{x} = \delta_{il}\delta_{jm}\delta_{kn}, \tag{24}$$

when integrated over the volume of the cell at level l. Clearly all seven octree basis functions within a given octree cell are mutually orthogonal. It is easy to see, given that these seven functions are orthogonal to the p_{000} Legendre block at the same level, that all octree basis functions, no matter what octree cells they occupy, are mutually orthogonal. The functional forms of the octree basis functions given in the equation above are particularly simple, but they are not unique. Alternative functions can be generated by using any 7×7 orthonormal matrix, as in equation (4) to produce a new set.

It is not difficult to see that if a given realization of a Gaussian white noise field is expanded using the p_{000} Legendre blocks at both level l and $l+1$, then the expansion coefficient of each block at level l is just the sum of the corresponding eight coefficients of its child cells. This relationship between the parent and child coefficients is identical to that used by both Pen (1997) and Bertschinger (2001) for refining a real-space Gaussian white noise field.

From a coding point of view, it is tempting to add Panphasia to GRAFIC2 using only the information in S_1 block coefficients as this would be quick and easy to do. However, as we will see in the tests shown later in this paper that using the S_1 block alone does a poor job in reconstructing the Panphasia phase information, so we cannot recommend this approach.

We now define a set of S_8 octree basis functions. These are built from the eight Legendre block functions which are products of the zeroth and first Legendre polynomials. We can do this in an analogous fashion to equation (23) by first defining a set of four one-dimensional functions, E_i, that are the S_8 analogues of the two D_0 and D_1 functions used to define the S_1 octree basis functions. These four functions are built from combinations of the P_0 and P_1 Legendre polynomials. Similarly for S_{27} we would need six functions built from P_0, P_1 and P_2.

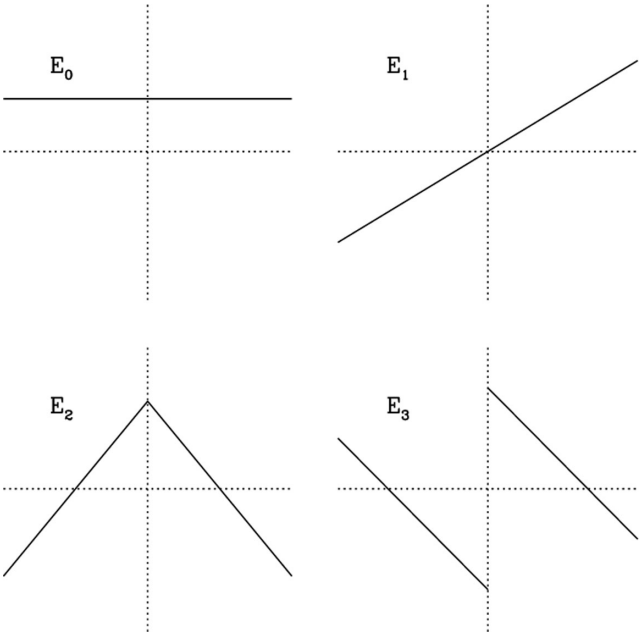


Figure 2. The functions defined by equations (25)–(28), where u is plotted on the horizontal axis. These four one-dimensional functions can be used to construct a set of S_8 octree basis functions as explained in the main text.

The set of four E_i functions consists of the pair with the functional forms of the P_0 and P_1 Legendre polynomials plus a pair of functions that have discontinuities about the origin. The left and right halves of the latter pair are linear combinations of P_0 and P_1 Legendre polynomials, scaled to half the width, and one is even and one odd about the origin. The four functions are as follows:

$$E_0(u) = \begin{cases} 1 & \text{if } -1 \le u < 1; \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

$$E_1(u) = \begin{cases} \sqrt{3}u & \text{if } -1 \le u < 1; \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

$$E_2(u) = \begin{cases} \sqrt{3}(1-2u) & \text{if } 0 \le u < 1; \\ \sqrt{3}(1+2u) & \text{if } -1 \le u < 0; \\ 0 & \text{otherwise.} \end{cases} \tag{27}$$

$$E_3(u) = \begin{cases} (2-3u) & \text{if } 0 \le u < 1; \\ -(2+3u) & \text{if } -1 \le u < 0; \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

These functions are shown in Fig. 2. With respect to the origin, the E_0 and E_2 are even functions and E_1 and E_3 are odd functions. By construction, all four functions are orthogonal and obey a normalization condition:

$$\int_{-1}^{1} E_i(u)E_j(u)\mathrm{d}u = 2\delta_{ij}. \tag{29}$$

We combine these functions to create an analogous set of three-dimensional functions to equation (23) to give

$$G^l_{ijk}(\boldsymbol{x}) = \frac{1}{\Delta_l^{3/2}} E_i\left(\frac{2x_1}{\Delta_l}\right) E_j\left(\frac{2x_2}{\Delta_l}\right) E_k\left(\frac{2x_3}{\Delta_l}\right), \tag{30}$$

where i, j, k are the integers either zero, one, two or three, and x_1, x_2, x_3 are the Cartesian components of $\boldsymbol{x}$ and it is assumed that the origin is the centre of the octree cell at level l. The functions are mutually orthogonal and obey a normalization condition:

$$\int G^l_{ijk}(\boldsymbol{x})G^l_{lmn}(\boldsymbol{x})\mathrm{d}^3\boldsymbol{x} = \delta_{il}\delta_{jm}\delta_{kn}, \tag{31}$$

where the integral is over the volume of the level l octree cell. There are 64 functions in total. The eight functions defined by the values of i, j and k all being zero or one are the S_8 Legendre block functions at level l. The functional forms of the remaining 56 functions are those of the S_8 octree basis functions. All of the basis functions have a least one discontinuity in value or slope about the principal coordinate planes defined by $x_1 = 0$, $x_2 = 0$ and $x_3 = 0$. All 64 functions can be built from combinations of S_8 Legendre blocks placed in the eight level $l + 1$ child cells.

For practical applications, it is easier to work with the smaller set of eight Legendre block functions rather than the 56 distinct octree basis functions. The actual S_8 basis functions used for Panphasia are defined in terms of Legendre blocks in Appendix A.

4 CHOOSING A SUITABLE PSEUDO-RANDOM NUMBER GENERATOR

Once a set of octree basis functions has been chosen, the next step for creating a realization of a Gaussian white noise field is to assign a value drawn from a Gaussian probability distribution to each octree basis function. This requires choosing a pseudo-random number generator and establishing a mapping between the linear pseudo-random sequence it produces and the octree basis functions. We will discuss the mapping first as the requirements of the mapping drive the choice of pseudo-random generator.

The octree functions form a four-dimensional discrete space. For a given choice of building block functions, there will be a fixed number of octree functions per octree cell. We can develop a mapping as follows:

(i) first, establish an ordering of the different types of octree basis functions belonging to each octree cell;

(ii) secondly, an ordering of the octree cells at a given level of the octree using a raster scan pattern over the three physical dimensions of the octree;

(iii) finally, an ordering by octree level starting at the root node and descending.

We give the full details of the ordering used for Panphasia in Appendix B.

In general, a randomly chosen point in the root cell will overlap an infinite number of basis functions. The values of the expansion coefficients of these functions are determined by an infinite series of short segments of the pseudo-random sequence which are spaced progressively further and further apart as the octree is descended. If the cost of accessing these coefficients were proportional to the linear separations between these segments, it would be impossible in practice to descend far from the root cell. This would be a major limitation for the method.

To avoid this limitation, we need a generator that allows essentially random access to the entire sequence at a reasonable computational cost. The ability to jump N places in at worst of the order of $\log N$ time is highly desirable as this opens up the possibility of using the entire period of the generator.

There are ways of accessing a pseudo-random sequence that are independent of the jump size. This can be done using an encryption algorithm which takes as input the linear position on the pseudo-random sequence and then scrambles the position in a highly non-linear way to produce a pseudo-random number. This kind of calculation is typically quite expensive if the aim is to produce cryptographically strong pseudo-random numbers. However, it is possible to produce a relatively fast pseudo-random number generator with this approach, albeit one that should probably not be used for encryption. The RAN4 random number generator from Press et al. (1992) is an example of this type. The authors found that the RAN4 generator produced good quality random numbers for sequences of a billion numbers. While we used this generator for a prototype to Panphasia, this particular generator is not suitable for our purposes as the total sequence length is too short. In principle, it ought to be possible to devise a generator along similar lines to RAN4 but with a longer sequence.

Rather than taking this route, we decided instead to take a generator that has been described in the literature and that has been in common use and well tested. Following a recommendation,[1] we have used a generator first published in L'Ecuyer, Blouin & Couture (1993). The generator has several names in the literature, but we will call it MRGK5-93. This generator is available as part of the GNU scientific library[2] where it is called GSL_RNG_MRG.

The internal state of MRGK5-93 can be represented as a five-element column vector with each element being an integer in the inclusive range 0, $m-1$, where m is the prime $2^{31}-1 = 2,147,483,647$. Given the nth state, T_i^n, the next internal state is generated by a matrix operation: $T_i^{n+1} = M_{ij}T_j^n \bmod m$, where modular arithmetic, base m, is applied to the results of the matrix multiplication. A uniformly distributed pseudo-random number between zero and one is associated with each state:

$$r_n = \begin{cases} \frac{T^n(1)-1/2}{m} & \text{if } 0 < T^n(1) < m; \\ \frac{m-1/2}{m} & \text{if } T^n(1) = 0. \end{cases} \tag{32}$$

The matrix for advancing one step for MRGK5-93 is

$$M_{ij} = \begin{pmatrix} a_1 & 0 & 0 & 0 & a_2 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \tag{33}$$

where $a_1 = 107,374,182$ and $a_2 = 104,480$. The state vector for the starting point of the sequence used for Panphasia together with a few other examples of state vectors is given in Table B1.

Although cast as a matrix multiplication above, the operations required to advance the generator one step can be implemented very efficiently on a computer. Because matrix multiplication is associative and the property of associativity is not affected by the application of modular arithmetic base m to matrix multiplication, the jump matrix which advances the state N steps at a time is just $M^N \bmod m$. The use of modular arithmetic ensures that the 25 coefficients of this matrix always remain in the range 0, $m-1$. The cost of advancing the state vector with a general matrix of this form is typically 30 times more expensive than advancing by a single step. The cost of building a jump operator, $M^N \bmod m$, starting from equation (33) is $O(\ln N)$ which makes it practicable to assign virtually the whole pseudo-random sequence to the octree basis functions.

From a computational point of view, the cost of evaluating the properties of a single octree cell in Panphasia is still very expensive as it requires the evaluation of typically thousands of pseudo-random numbers at many different places in the sequence. However, the access pattern needed for making initial conditions is highly localized so the actual number of pseudo-random numbers that need to be evaluated per cell is close to just 10. Using a raster scan access pattern for the cells which mirrors the mapping of the pseudo-random sequence on to the basis functions minimizes the number of large jumps required and maximizes the number of consecutive accesses to the sequence. Further improvements in speed can be obtained by caching the results of previous evaluations. With current processors it typically takes about 2 μs on average per cell to return the expansion coefficients belonging to that cell. There is still scope to improve the speed of the code with further optimizations, but in practice the generation of the white noise field typically takes only about 20 per cent of the time to generate initial conditions, excluding the I/O.

As stated in the documentation for the GNU scientific library, the full period of the MRGK5-93 generator is $P_{\rm gen} = m^5 - 1 \simeq 4 \times 10^{46}$. This means that the period consists of every possible state vector with the exception of the null vector. Interestingly, the generator has a subperiod, $P_{\rm sub} = (m^5-1)/(m-1) \simeq 2 \times 10^{37}$. Over multiples of this subperiod, the jump matrix becomes a multiple of the identity matrix (Booth, private communication). When this multiple (which ranges from 1 to $m-1$) takes on small values, there are significant correlations between pseudo-random numbers at this precise separation. The large size of the subperiod precludes any possibility of such a coincidence occurring for levels shallower than 40 in the octree. Going deeper still in the tree, there is a remote possibility that some pseudo-random numbers separated by a multiple of $P_{\rm sub}$ may occur in a given set of initial conditions. The chances of this happening, however, are vanishingly small for any randomly chosen region.

Of more concern is the more general question of whether this generator provides sufficiently good pseudo-random numbers for making cosmological simulations. It is desirable that the generator passes a diverse set of randomness tests. However, even in the epoch of precision cosmology the requirements on a generator for cosmological simulations are less strict than many other applications such cryptography. Some deviation from randomness is acceptable for cosmological initial conditions provided it is sufficiently small. The fact that the pseudo-random numbers are discrete and not truly uniformly distributed is not a concern, although as described in Appendix B we do take measures to mitigate the discreteness effect to ensure that the tail of the Gaussian distribution for the Gaussian pseudo-random numbers is properly populated.

For some purposes such as encrypting secret messages or for gambling machines, it is highly desirable that the pseudo-random sequence cannot easily be predicted by studying a small part of its output sequence. In this respect, MRGK5-93 performs poorly as just five consecutive numbers are sufficient to deduce all five elements of the state vector. Once the state vector is known, the whole sequence is determined. This property means that an n-tuple (for $n > 5$) of pseudo-random numbers produced by MRGK5-93 cannot uniformly sample the n-point joint probability function. However, the coefficients, a_1 and a_2, used in the generator were selected in part by a requirement that the deviations from uniformity in the n-point joint probability function for $6 \le n < 21$ are confined to

[1] The author is grateful to Stephen Booth of the Edinburgh Parallel Computing Centre for suggesting a suitable generator and providing him his own f90 implementation of the generator.

[2] http://www.gnu.org/software/gsl/

very fine scales (L'Ecuyer et al. 1993). In practice, this means that detecting deviations from randomness in the joint n-point function requires very large samples in order to distinguish non-uniformity from shot noise. This small-scale non-uniformity is not obviously a problem for cosmological initial conditions as any effect is likely to be dwarfed by sources of numerical error in any actual N-body simulation.

A second feature of MRGK5-93 is that over the entire period, with one exception, each pseudo-random number appears m^4 times exactly – a highly subrandom pattern. So a test looking at the frequencies of occurrence of different pseudo-random numbers will show a failure for a sufficiently large sample of the sequence. This deviation from randomness is extremely small so hardly a concern for making initial conditions. Given these known limitations, it is interesting to see how well MRGK5-93 does perform in wide range of standard tests of randomness.

L'Ecuyer & Simard (2007) have developed a software library, TESTU01, for testing pseudo-random number generators empirically. The results of libraries of tests performed on many common generators, including MRGK5-93, are given in this paper. Their most rigorous test battery, called BigCrush, yields 160 independent statistical results from a total of 106 separate tests. A total of 2.7×10^{11} pseudo-random numbers for each generator are used in these tests. The outcome of each test is a p-value, which for a perfect generator would be expected to be drawn from a uniform distribution between 0 and 1. A generator is said to fail a test if the p-value is within 10^{-10} or zero or unity. A test where the p-value is within 10^{-4} of the same limits is noted as suspect. Quite a few commonly used generators do fail multiple tests. The MRGK5-93 generator does not fail any of the BigCrush tests and none of the p-values are suspect.

This is encouraging and on this basis we are content to use this generator. The size of the sample tested is larger than would typically be used to generate most sets of initial conditions, although some like those of the MXXL require more. We do nonetheless expect this generator to eventually fail a randomness test applied to a larger sequence of numbers than tested by BigCrush for reasons explained above. The deviations from randomness we described are small and not a great concern. Tests of pseudo-random number generators, however, can never completely set the mind at rest as it is impossible to be sure that some new random test may reveal a significant flaw previously undetected.

5 ADDING PANPHASIA TO THE IC_2LPT_GEN CODE

In this section, we describe how to modify the IC_2LPT_GEN code for making cosmological initial conditions to use Panphasia to set the phase information. There are two main goals of this section. First, to show that it is possible to make initial conditions that accurately reconstruct the phases defined by Panphasia and secondly to act as a practical guide to help anyone wanting to add Panphasia phases to other initial condition generators.

5.1 Overview of IC_2LPT_GEN

The IC_2LPT_GEN code is used by the Virgo Consortium to generate Gaussian initial conditions for a variety of projects relating to large-scale structure, galaxy formation, the internal structure of dark matter haloes and the formation of the first stars. The code is able to make resimulation initial conditions with displacements and velocities calculated using second-order Lagrangian perturbation theory (2LPT). While the initial conditions we make for this paper use this feature, there is no significant interaction between the methods we describe here and those required to generate 2LPT initial conditions. We will describe only the most relevant features of the code here. A more detailed description of the code, including the method to make 2LPT multiscale initial conditions, can be found in Jenkins (2010).

When it comes to modifying the code to use the phase information from Panphasia, it is useful to divide the kinds of initial condition that IC_2LPT_GEN can make into two classes. The first class is cosmological initial conditions where a single Fourier grid is used to generate displacement and velocity fields for all the particles in a large periodic domain. An example of this is the MXXL simulation (Angulo et al. 2012), which modelled 303 billion particles in a cubic volume of about 4 Gpc on a side.

The second class, which we will call resimulation initial conditions (or equivalently zoom simulations), uses multiple Fourier grids to compute the displacement and velocity fields to generate multiscale initial conditions. In common with cosmological initial conditions, a grid is used that covers the entire simulation domain. We will call this the parent grid or outer grid. The extra grids, which we will call collectively the inner grids, typically have a similar number of grid points as the parent grid, but are physically smaller and placed concentrically around a focal point of interest in the simulation volume. The IC_2LPT_GEN code can place many nested subgrids around a point allowing it to make very high resolution initial conditions for the region that is contained within all of the grids. The remainder of the simulation volume is required to provide the appropriate tidal forces only and is modelled at lower mass resolution to reduce computational cost. A recent example of resimulation initial conditions made by IC_2LPT_GEN is those created for the Phoenix project (Gao et al. 2012), the computational goal of which was to model at high numerical resolution the dark matter in individual galaxy clusters, selected from the Millennium simulation (Springel 2005).

The IC_2LPT_GEN code uses Fourier methods to generate the Gaussian displacement and velocity fields for each grid. The fluctuations are created in k-space by generating independent random amplitudes and phases for each Fourier mode (subject to the condition that the field is real) with the option of using one of several different pseudo-random number generators to calculate a series of independent Gaussian pseudo-random numbers. The amplitudes of the Fourier modes are scaled appropriately so as to reproduce the desired linear density fluctuation power spectrum. For a given grid, the Fourier modes are set only in a range between low-k and high-k limits.

For cosmological initial conditions utilizing a single grid, these limits in k are determined by the fundamental mode of the simulation cube at low k and typically by the particle Nyquist frequency at high k. The high-k cut-off is chosen to be spherical so that the fluctuations are isotropic at small scales. For resimulation initial conditions, the low-k and high-k limits associated with each nested grid in real space are dovetailed to ensure that the power spectrum in the high-resolution region has the appropriate contributions all the way from the fundamental mode of the simulation cube down to the particle Nyquist frequency of the high-resolution region.

All grids are treated in the same way by IC_2LPT_GEN which means that all field quantities are periodic on the physical scale of the grid. This periodicity is only strictly correct for the parent grid. However, the effect of periodicity on the other grids can be limited by choosing a low-k cut-off which is significantly larger than the fundamental mode of that grid. If this condition is met, then the

correlation length of the field on the grid is much smaller than the size of the grid itself.

Once the displacement and velocity fields (and other fields) have been calculated on a grid, the IC_2LPT_GEN code uses interpolation to compute the values of these fields at the locations of unperturbed particles. The IC_2LPT_GEN code uses the value of the fields and their spatial derivatives in the interpolation from the grid points to the positions of the particles in the unperturbed particle load. The interpolation scheme used is described in detail in the appendix of Jenkins (2010).

For a resimulation to be successful, the phases present in the original cosmological simulation must be reproduced on the parent grid. For a resimulation with higher resolution in some region of interest than the parent simulation, additional high-k power is needed. The choice of the phases for this extra power is not constrained by the large-scale power. In IC_2LPT_GEN the phase information for the high-k power is determined by a series of arbitrarily chosen pseudo-random number seeds: one for each extra grid. The precise positioning and dimensions of the extra grids are ill determined: all that is required is that it be approximately concentric about the region of interest. All of these freedoms contribute to making the choice of the phase information at high-k power rather arbitrary. In practice, the phase information is encoded in a text parameter file, the length of which increases as the number of grids increases.

5.2 Adding cosmological initial conditions from Panphasia to IC_2LPT_GEN

The phase information for cosmological initial conditions is most compactly represented as a finite set of amplitudes and phases, each associated with a periodic plane wave that spans the simulation volume. These waves range in wavelength from the fundamental modes of the periodic simulation volume to a cut-off wavelength typically determined by the interparticle spacing. This contrasts with the equivalent octree basis function representation, where in order to reproduce the phases of these particular waves exactly, an infinite number of expansion coefficients are needed. Nonetheless, it is possible to accurately reproduce the phase information with a finite number of octree basis functions, particularly when using the S_8 octree functions introduced in Section 3. In practice, only octree basis functions down to some maximum depth, $l_{\rm max}$, in the octree can be used, where the value of $l_{\rm max}$ should be as large as possible given limited resources.

The IC_2LPT_GEN code makes initial conditions using Fourier methods starting with a k-space representation of a Gaussian field on a cubic grid. The natural way to incorporate Panphasia is to choose a Fourier grid which is commensurate with the Legendre block expansion of Panphasia at level $l_{\rm max}$ of the octree.

We assume that a particular cubic region within Panphasia consisting of N^3 whole octree cells at level l has been selected to define the phase information for a given cosmological simulation. The corresponding dimension of the cubic Fourier grid, M, should obey $M = 2^{(l_{\rm max}-l)}N$, where $l_{\rm max} \geq l$. This ensures that there is a one-to-one correspondence between grid points and octree cells at level $l_{\rm max}$. The use of fast Fourier transform algorithms places restrictions on the value of N, limiting it to be a product of small prime factors only.

Having matched the grid to Panphasia, the IC_2LPT_GEN code takes only the information provided by Panphasia as deep as level $l_{\rm max}$. For each octree cell at level $l_{\rm max}$, we know the basis function coefficients of the expansion of Panphasia for S_8 Legendre blocks. We can associate a separate grid with each of the eight types of Legendre block, and assign an expansion coefficient to a corresponding grid point for all cells. Each of these grids is an independent discrete realization of a Gaussian white noise field. The remaining task is to combine the information on these eight fields to produce a single field on a grid that accurately reconstructs the Panphasia phases.

Taking the grid points to represent delta functions, scaled by the corresponding values of the expansion coefficient, we can in principle exactly regenerate Panphasia truncated to level $l_{\rm max}$ by convolving each of the eight grids with the appropriate Legendre block and co-adding the results to give a single continuous field. This continuous field would nonetheless have a discrete representation in k-space because of periodic boundary conditions.

In practice, in IC_2LPT_GEN the eight grids are combined in k-space to give a discrete combined field. This is done by applying a fast Fourier transform to each real grid to produce a k-space equivalent. Once in k-space, the convolution with the appropriate Legendre block is achieved by multiplying by the Fourier transform of the Legendre block (given by equation 15). An additional phase factor corresponding to a uniform translation in real space has to be included in this convolution for IC_2LPT_GEN. This is because IC_2LPT_GEN places a grid point at the coordinate origin, which means that the grid points and the octree cell centres are everywhere displaced from each other by half a grid spacing in each of the Cartesian directions. The translation by half a grid spacing in all three Cartesian directions is needed to ensure that the phase pattern appears in the correct physical location. Summing the eight fields produced by the convolution results in a discrete and bandwidth-limited representation of Panphasia in k-space. This field is not a true Gaussian white noise field as the S_8 Legendre blocks placed at a given level of the octree are not a complete basis set.

The field produced in this way can be restored to a true white noise field by adding an additional uncorrelated field with an ensemble averaged power spectrum:

$$P^{l}_{\rm add}(\boldsymbol{k}) = 1 - \sum_{\rm S_8} j^2_{i_1 i_2 i_3}(\boldsymbol{k}\Delta_l). \tag{34}$$

Using identity (18) it is easy to see that the power spectrum of the truncated field averaged over all directions deviates as k^4 from a white noise field at large spatial scales for the Legendre block functions of S_8 and as k^2 for the S_1 Legendre block.

This extra component should ideally be generated from the basis expansion coefficients of Panphasia for levels deeper than $l_{\rm max}$, but there has to be a cut-off in practice and we take it to be $l_{\rm max}$. So instead IC_2LPT_GEN generates a ninth independent white noise grid and uses the form of $P_{\rm add}(\boldsymbol{k})$ so that when combined with the other eight fields the result is a true white noise field. This guarantees that the initial conditions have the correct power spectrum with an isotropic cut-off in k, but comes at the price that the phases of Panphasia particularly at small scales are not perfectly reconstructed. The pseudo-random numbers needed for the ninth grid are not part of Panphasia, but as explained in Appendix B the coefficients for the ninth field are generated at the same time as the basis coefficients for the other eight grids.

The requirement to build the field with so many components potentially has some negative practical effects on the memory efficiency of the code. The memory requirements would be significantly increased if all nine grids had to be stored at the same time when using the coefficients for all eight Legendre blocks. In fact, the code already needs to be able to store four grids in order to make 2LPT resimulation initial conditions as described in Jenkins (2010). These grids however are not required until the white noise field has been

computed. So there does not need to be any increase in the memory usage of the code provided the white noise field is evaluated three times in succession. In practice, the actual code stores five grids and therefore has to evaluate the white noise field twice. As it takes both more cpu time and more memory to make initial conditions using S_8 in preference to S_1, there has to be a good reason to prefer it. This applies even more strongly to S_{27} which would require 28 grids to be combined. To keep the memory requirements the same with S_{27} would require evaluating the white noise field six times, and each evaluation of the white noise field would be significantly more expensive too. There would need to be an even stronger reason for rejecting S_8 before considering the S_{27} set.

The ultimate decision as to the best set of octree basis functions is made later in the paper from studying the end states of simulations. It is useful nonetheless as a guide to help interpret the results of these simulations to compare the different choices of the octree basis function by looking at linear density fields first. We define first an error field, which is the difference between the linear density field reproduced by the method and the true Panphasia linear density field. We can then characterize this error field by its power spectrum. Because the ninth field is independent of Panphasia, the error power spectrum is simply $2P_{\rm add}$.

Using this error power spectrum we can quantify the effect of this error field on density fluctuations in the initial conditions, for some assumed power spectrum $P(k)$, by determining the fractional error in the rms fluctuations, $\epsilon_{\rm rms}$, as a function of volume as

$$\epsilon^2_{\rm rms} = \frac{\int 2P_{\rm add}k^2\,W^2(k){\rm d}k}{\int P(k)k^2\,W^2(k){\rm d}k}. \tag{35}$$

The top and bottom terms on the r.h.s. are the rms fluctuations in the error field and the true field, respectively, smoothed by a suitable spherical filter, W, with some characteristic scale that can be varied. For spherical perturbations, the collapse epoch is determined by the linear overdensity at some fiducial epoch, so these fraction rms fluctuations give an indication of how structure formation is affected as a function of scale.

In Fig. 3, we plot $\epsilon_{\rm rms}$ against the effective volume of the filter, W, for power-law initial conditions with $P(k) \propto k^{-2.75}$ made with either the S_1 or S_8 Legendre blocks. We take W to be a Gaussian filter with the zeroth and second moments matched to a spherical tophat with a volume plotted on the x-axis in units of the volume per grid cell. The tophat filter itself is unsuitable because the top integral in equation (35) is dominated by a surface term rather than by the volume term for the error field generated by S_8. The choice of the power-law index is appropriate for CDM initial conditions with fluctuations populated down to a particle Nyquist frequency corresponding to a particle mass of about $10^6\,{\rm M}_\odot$ which will be featured in the tests later in this section.

Clearly the field generated by S_8 is a much better approximation to Panphasia for all volumes than S_1, with the difference increasing with increasing volume. For a fractional error of $\epsilon_{\rm rms} = 0.01$, there is a factor of about three orders of magnitude in volume between the S_1 and S_8 initial conditions. This means that to achieve the same accuracy in the phase reconstruction using initial conditions made using the S_1 octree basis functions requires orders-of-magnitude higher numerical resolution than for S_8. In Section 6, we will see how these differences translate to the end states of simulations. Before this, however, we consider the task of how to adapt IC_2LPT_GEN to make resimulation initial conditions.

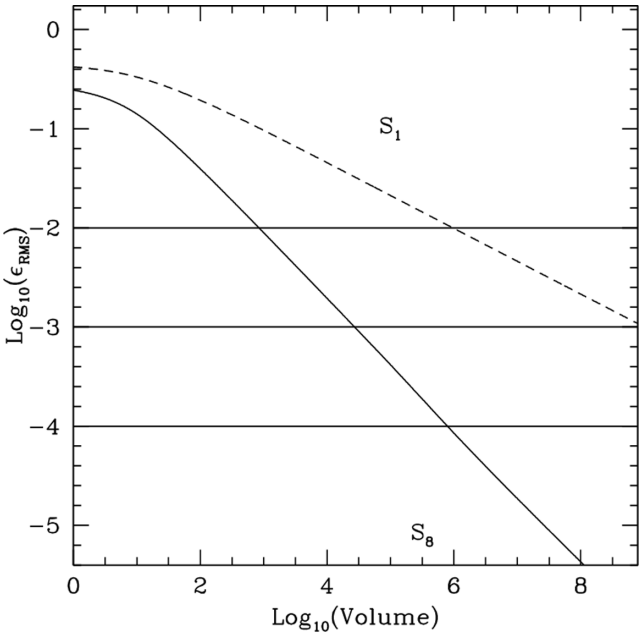


Figure 3. A comparison of the relative accuracies in reproducing the correct phases in initial conditions for two choices of sets of Legendre blocks. The quantity of the y-axis, defined in equation (35), is plotted as a function of the volume of the smoothing filter in units of the volume of the grid cells. The integrals have been truncated at the grid Nyquist frequency.

5.3 Adapting IC_2LPT_GEN to make resimulation or zoom simulations using Panphasia

As described earlier, the IC_2LPT_GEN code generates what we will call Fourier resimulation initial conditions in a piecewise fashion by calculating the displacement fields on a series of nested grids about some focal point of interest in the parent simulation. The total linear displacement for any particle is just the sum of the linear displacements generated by the individual grids that it spatially overlaps. The displacement fields for each grid are generated in the same way as for cosmological initial conditions with starting point being to generate a k-space representation of a Gaussian field on a grid. To adapt IC_2LPT_GEN to use Panphasia requires finding a way to generate equivalent k-space Gaussian fields based on Panphasia. Once this has been achieved, no further modification of the code is needed to produce initial conditions.

The precise placement of the Fourier grids in IC_2LPT_GEN Fourier resimulation initial conditions is largely arbitrary. All that is required is that the different grids be nested and centred approximately on the region of interest. With the Fourier method the displacement field pattern is tied to the grid and would move rigidly with the grid if it was decided to place the grid in a slightly different location. In contrast, the octree basis functions of Panphasia have fixed coordinates so moving the grid will not lead to the phase pattern itself shifting. In practical terms this means that it is not necessary when using Panphasian phases to publish the grid positions. It is true that the 'error fields', that is the differences between the true initial conditions and those actually generated, will depend on the choice of grid positions, but as the aim in numerical work is to minimize the numerical effects to the point where they do not affect the scientific conclusions, it should not be necessary to specify the precise grid positions.

As with cosmological initial conditions, the extra grids must be arranged so that the grids are commensurate with the octree cells which means there is only a discrete set of positions where the grids can be placed. These positions are those where the grid points are located at the corners of octree cells. The sizes of the grids measured in grid cells are also restricted for computational and numerical reasons, by both the method of parallelization in IC_2LPT_GEN and through the use of fast Fourier transforms. In practice, the ratios of the physical sizes of the nested grids have to be simple fractions, and for all the tests in this paper they differ by powers of 2 only.

In the Fourier method for making resimulation initial conditions, the phase information on each grid is independent from that on every other grid simply because each Fourier mode is set with a different pseudo-random number and each Fourier mode belongs to a single grid. To make the grids independent using Panphasia, all that needs to be done is to assign each octree basis function to a single grid. Any alternate approach that allows an octree basis function to be split between grids does not appear attractive as the contributions on the different grids would be sampled on different scales and, because they are coherent, add in a complex way with associated fringing.

While the octree basis functions themselves are perfectly confined to a single cell, once a convolution has been applied to generate the initial conditions, the information contained within a single cell is propagated to all points on the grid, although in a far from uniform way. As discussed in Appendix A, both the zeroth and first moments evaluated about the cell centre of the octree basis functions are identically zero. This vanishing of the zeroth and first moments means that the information stored in the octree basis function is more strongly localized than for the white noise as a whole occupying the same volume. This property of partial locality makes the size of edge effects when making multiscale initial conditions using Fourier methods smaller than might first be supposed.

As the octree basis functions are built out of Legendre blocks, the actual mechanics of calculating the k-space Gaussian field for resimulation initial conditions is the same as for cosmological initial conditions. The only extra ingredient needed to make Panphasia resimulation initial conditions is to decide how to partition the octree basis functions between a given set of nested grids. While the majority of octree basis functions only overlap the outer most grid so there is no choice, for all other octree functions there is a choice between two or more grids. The choice should ideally be the one that gives the best initial conditions, although it is difficult to be precise about exactly what best means in this context. We have opted to adopt a heuristic approach to the solution of this problem. The justification of this approach is that it can be demonstrated to work well in practice as we show later. This heuristic approach can be described by four rules.

(i) Rule 1: *'Any octree basis function that can be included, should be included'*. As every octree basis function contributes to the phase information, missing out any of the octree basis functions will degrade the quality of the phase reconstruction. This rule makes it easy to count basis functions that must be used in total over all grids and makes it simple to check in the code that all have been placed.

(ii) Rule 2: *'With the exception of the outermost grid, only whole octree basis functions can be assigned to a grid.'* The reason for this rule is to minimize unwanted edge effects. As explained earlier in this section, in IC_2LPT_GEN all of the grids are periodic. These are the correct boundary conditions only for outermost grid, and it is therefore necessary for all other grids to try and minimize the edge effects due to periodicity. Only whole octree basis functions are guaranteed to have vanishing zeroth and first moments and this is required to limit their range of influence as discussed earlier in this subsection. The requirement that whole octree basis functions are placed in these grids puts further restrictions on the possible sizes and precise placements of the inner grids. The smallest octree basis function measures two cells along each edge of the grid, which means both that the grid dimensions must be even and that the edges of the grids must line up with the octree cells these basis functions occupy.

(iii) Rule 3: *'Except where rule 4 is broken, an octree basis function should be placed on the innermost grid allowed by rule 2.'* Far from any boundary it is desirable that an octree basis function is represented over as many grid cells as possible. This is because the Fourier transform of the octree basis functions is not intrinsically bandwidth limited but its representation on a grid will be bandwidth limited. Placing a basis function over as many cells as possible minimizes the truncation of power at small scales. Near a grid boundary however there is another factor to consider which requires rule 4.

(iv) Rule 4:*'For grids other than the outermost, and for octree basis functions other than the smallest, no octree basis function should be placed within a perpendicular distance, measured from its edges, from the grid boundary that is less than a factor X times its own edge size.'* A rule of this kind is required to minimize unwanted edge effects due to periodicity for all but the outermost grid. The reason for not wanting to place large octree base functions close to a grid edge is simply that the larger an octree basis function, the further its influence spreads in the initial conditions. This requirement is in direct opposition to rule 3 which encourages larger octree cells where possible. The use of a distance criterion with a factor 'X' to be determined empirically ensures that a compromise is possible. For scale-free initial conditions, it would be natural to take 'X' to be constant in the absence of any characteristic scale other than the cell size itself. For CDM models, for example, the power-law index varies with scale but at small scales is slowly varying. As there is a smallest octree basis function that can be placed on a grid, and rule 1 requires these smallest basis functions to be present, they have to be treated exceptionally and are allowed to be placed right up to the boundaries.

Once the factor X in rule 4 is decided, there is then a unique solution for placing the octree cells on a given set of grids. Different values of X do in some cases result in the same placement. Small values of X mean that the octree basis functions close to the edges of the inner grids are large and therefore have longer range effects. Large values mean that octree basis functions are assigned to small numbers of grid points and are therefore less accurately represented. As part of the next section we will compare dark matter haloes generated with different values of X to determine a close to optimal value. A code which implements the rules above is included in addition to Panphasia in the public code release. See Jenkins & Booth (2013).

6 TESTING THE ACCURACY WITH WHICH THE PHASE INFORMATION FROM PANPHASIA CAN BE REPRODUCED

The main goal of this section is to demonstrate that it is possible to make good quality resimulation initial conditions using the phase information provided by Panphasia. To do this, we will test how well the methods for making cosmological and resimulation initial conditions described in the previous section succeed in this task. We

will judge success by looking at the end states of a set of simulations at redshift zero. A further goal of this section is to provide a guide to help anyone wanting to add Panphasia to an initial conditions code for how to test that the code is actually working.

Judging what constitutes 'good quality' resimulation initial conditions is inevitably rather subjective. There is relatively little published about the efficacy or otherwise of resimulations that can help define a standard. There is none at all that can be directly compared with the results of this section. This is because published methods of setting the phases do not define the phase in an objective way, so it is not possible to make the same initial conditions in two different ways and expect them to be the same, which is what we aim to do here. Nonetheless, it is reasonable to require that the size of the errors we can determine in this section should be comparable to or ideally smaller than the sizes reported in the literature for typical applications of resimulations. Hard numbers can be found in only a few published papers. These numbers show how well the bulk properties of dark matter haloes are reproduced for haloes resimulated at several numerical resolutions. As two simulations of the same halo at different resolutions do not have identical phase information, there is no reason to expect that the bulk properties should be exactly reproduced, but it is found for haloes represented by millions of particles or more that properties such as virial mass or maximum circular velocity are reproduced at sub-per cent accuracies (Springel et al. 2008; Hahn & Abel 2011; Gao et al. 2012). We will use this crude measure to judge whether the resimulation initial conditions produced from Panphasia are of a comparable standard to published methods or not, and declare them to be of good quality if they are.

Ideally, we need a reference calculation that precisely reproduces the phases from Panphasia. We have seen from Section 5.2 that this cannot be done, but it is possible in principle to produce a very accurate approximation to the true phases by using an extremely large Fourier transform to make cosmological initial conditions. We can then run a simulation using these initial conditions and create a reference end state. This can be compared to the end states of simulations starting from cosmological and resimulation initial conditions made using Fourier transforms of a size that would be used in practice because they can be readily afforded.

The plan for this section is as follows: in Section 6.1 we introduce the reference calculation; in Section 6.2 we describe and test a method to measure how well the phase information is reconstructed; in Section 6.3 we apply this measure to cosmological initial conditions; in Section 6.4 we demonstrate that the proposed resimulation method works well; finally in Section 6.5 we investigate the sensitivity of the method to changes in parameters such as the Fourier grid and the X parameter introduced at the end of the last section.

6.1 The simulations

We have chosen a fairly typical case of the resimulation method which is to resimulate a single isolated dark matter halo with a mass similar to the inferred mass of the Milky Way (MW; Springel et al. 2008; Stadel et al. 2009). We have chosen a halo from a completed high-resolution N-body simulation run by the Virgo Consortium called DOVE. This ΛCDM dark matter only simulation is a 70.4 Mpc h^{-1} periodic box with similar mass resolution to the Millennium-II simulation (Boylan-Kolchin et al. 2009). The cosmological parameters however differ from the Millennium-II and are listed in Table 1. These parameter values are taken from table 1 of Komatsu et al. (2011) and are based on constraints derived from the cosmic microwave background, BAO and the Hubble constant. The CDM transfer function for this model was calculated using CMBFAST (Seljak & Zaldarriaga 1996). The initial phases were taken from Panphasia and a 3072^3 Fourier grid was used to make the initial conditions. We give the precise location for the phases in Panphasia in Section 7.1. The simulation was run to redshift zero using the P-GADGET3 N-body code (Springel et al. 2008).

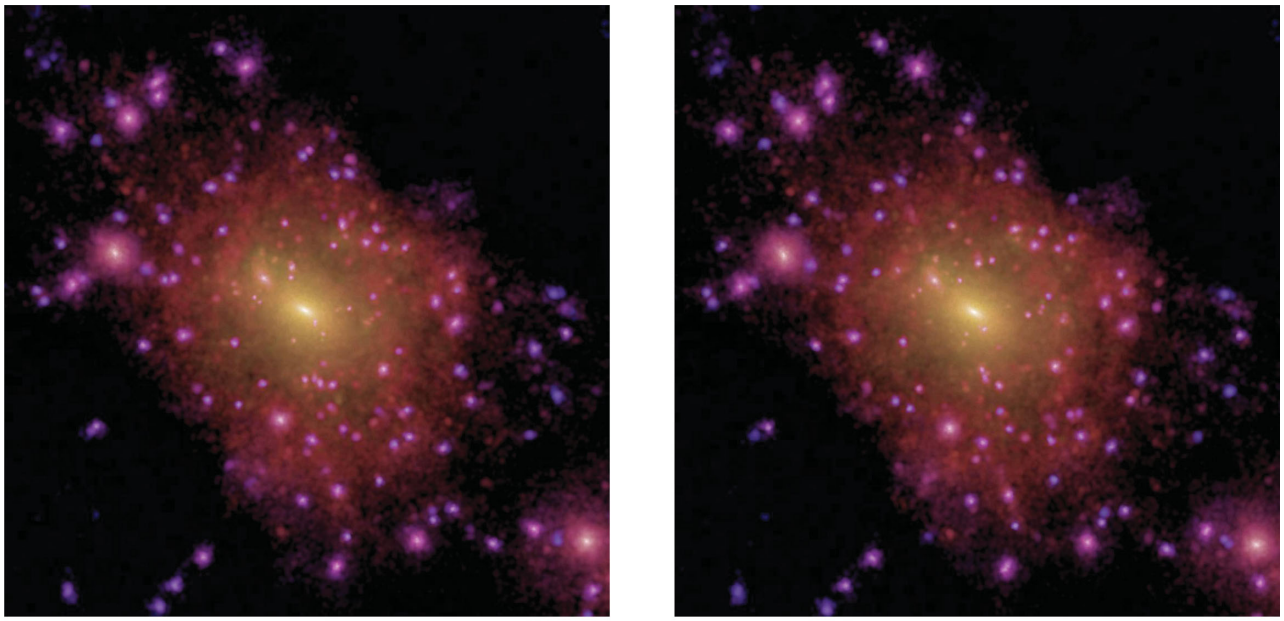
Table 1. The cosmological parameters of the DOVE simulation and for all of the test simulations in this paper: $\Omega_{\rm matter}$, Ω_Λ and $\Omega_{\rm baryon}$ are the average densities of matter, dark energy (with -1 equation of state) and baryonic matter in the model in units of the critical density; H_0 is the Hubble parameter; σ_8 is the square root of the linear variance of the matter distribution when smoothed with a tophat filter of radius $8\,h^{-1}$ Mpc; and n_s is the scalar power-law index of the power spectrum of primordial adiabatic perturbations.

Cosmological parameter	Value
$\Omega_{\rm matter}(z=0)$	0.272
$\Omega_\Lambda(z=0)$	0.728
$\Omega_{\rm baryon}(z=0)$	0.0455
H_0 (km s^{-1} Mpc^{-1})	70.4
σ_8	0.81
$n_{\rm s}$	0.967

The test halo was chosen by the author to have no close large neighbours by visual inspection of dot plots. Similar results to those presented in this section have been obtained with a second MW mass halo in a different part of the DOVE volume. The resimulation methods described in the last section have also been applied to resimulate cluster mass dark matter haloes in other Virgo simulations set up with Panphasian phases, and the properties of these haloes are reproduced to similar fractional accuracy as we report for the halo studied in this section. We can therefore judge the quality of the initial conditions by studying this one typical halo.

Although the DOVE simulation was used to select a halo, we will not use any data from the DOVE simulation in this paper. We have, however, compared the properties of the halo for the highest quality simulations in this paper with its counterpart in the DOVE simulation and find excellent agreement in its position and measured properties.

We expect that the accuracy with which the phase information is reproduced improves with the problem size as can be deduced from Fig. 3. To test the methods, we should therefore not aim to resimulate a halo at very high resolution. Going to the other extreme of low resolution would mean simulating a halo represented by just a few particles which would make it difficult to determine any of the halo properties due to the discreteness. As a compromise we have chosen a sufficiently high resolution for the halo to start to show the very rich substructure revealed by ultra-high-resolution simulations of dark matter halo formation (Springel et al. 2008; Stadel et al. 2009; Gao et al. 2012). The halo has about 250 000 particles within R_{200} and around 50 identifiable substructures with more than 20 particles as determined by the SUBFIND group finder (Springel et al. 2001).



(a) (b)

Figure 4. (a) Image of the reference halo at redshift zero. The projection is in the x_1-x_2 plane. The side of the image measures 1 h^{-1} Mpc. The intensity of the image is scaled by the integrated square density of dark matter, while the hue is set by the local velocity dispersion. See Springel et al.(2008) for more details about this method of making images. (b) A resimulation of the same halo discussed in Section 6.4.

We used the same software to build the particle load as the Aquarius haloes (Springel et al. 2008). The lowest mass particles are placed in a region that occupies a small fraction of the simulation volume. The particles making up the redshift zero halo and its immediate surroundings are located within this high-resolution region. Higher mass particles are placed further out around this region and provide the appropriate tidal field on the region of interest. The particle load has 787 939 high-resolution particles each with a mass of $6.24 \times 10^6\, h^{-1}\ \mathrm{M}_\odot$. The total mass within the DOVE volume is equivalent to about 1616^3 of these particles. The smallest Fourier transform that can be used to generate power down to the Nyquist frequency of high-resolution particles given the quantization constraints of Panphasia is 3072^3 – the same grid size used to make the DOVE initial conditions.

The displacements and velocities are calculated with 2LPT using the method in Jenkins (2010) for all of the initial conditions. The P-GADGET3 code was used to integrate the equations of motion from the starting redshift 63 to redshift zero. The same numerical parameters for P-GADGET3 were used for all simulations to ensure that any differences observed are due only to differences in the initial conditions. The gravitational softening comoving length for the high-resolution particles was $2\,h^{-1}$ kpc at all times. The SUBFIND group finder (Springel et al. 2001), which is integrated into P-GADGET3, was run on the high-resolution particles only. There are no heavier mass particles near the main halo at redshift zero.

For the reference calculation, we used a $12\,288^3$ Fourier transform to make a set of cosmological initial conditions. This required 35 Tb of RAM to run and was only possible through access to the new Distributed Research utilising Advanced Computing (DiRAC) II facility at Durham. Fig. 4(a) shows a projection of the reference halo made using the same software as was used to render the Aquarius haloes (Springel et al. 2008). The location and some of the bulk properties of the halo are both defined and given in Table 2.

Table 2. Some properties of the reference halo at redshift zero. The centre is the potential minimum. M_{200} and $M_{\rm vir}$ are masses within spheres centred on the potential minimum, with mean densities of 200 and 97 times the critical density, respectively. The radii of these spheres are R_{200} and $R_{\rm vir}$. The velocity dispersion, $V_{\rm disp}$, is that of the main subhalo as determined by SUBFIND. $V_{\rm max}$ is the maximum circular velocity, where all circular velocities are calculated assuming spherical symmetry about the halo centre. The halo spin is given by λ', first defined in Bullock et al. (2001). The spin is determined for all particles within a sphere of radius $R_{\rm vir}$ centred on the halo potential, $\lambda' = J/\sqrt{2}M_{\rm vir}R_{\rm vir}V_{\rm vir}$, where J is the angular momentum and $V_{\rm vir}$ is the circular velocity at the virial radius. The direction of the spin is given in terms of the directional cosines projected on to the (x_1, x_2, x_3) axes.

Property	Unit	Value(s)
Centre	Mpc h^{-1}	(43.1732, 49.9170, 2.2070)
M_{200}	$10^{10}\,h^{-1}\ \mathrm{M}_\odot$	91.61
$M_{\rm vir}$	$10^{10}\,h^{-1}\ \mathrm{M}_\odot$	110.02
R_{200}	kpc h^{-1}	157.95
$R_{\rm vir}$	kpc h^{-1}	213.47
$V_{\rm disp}$	km s^{-1}	102.72
$V_{\rm max}$	km s^{-1}	181.71
λ'	–	0.0133
$(c1, c2, c3)$	–	$(-0.213, -0.651, -0.728)$

6.2 How to measure convergence in the phase information

We need a quantitative measure to determine how well the phase information has been reproduced for different sets of initial conditions. As the true solution is not known, the best we can do is to use the reference halo as a close approximation to the truth, and compare the haloes produced from other sets of initial conditions

to the reference halo. This assumption cannot be fully tested but we will make some consistency checks in the next subsection to show that it is reasonable.

Rather than focusing on how well particular properties such as the virial mass are reproduced, we will measure how well the redshift zero particle positions are reproduced. As all simulations start with the same particle load, we can match the particles that originate from the same location of the particle load, for any pair of simulations, and measure their separations as a function of time. We expect the distribution describing the relative positions of particles in any pair of our simulations to diverge over time. If for the end state at redshift zero all the particle positions match well, then we can be confident that not only will the physical properties of the halo at redshift zero agree very well between the two simulations, but also this agreement will apply over the entire past history. For if this were not the case, it would require a vast coincidence to have occurred – something that we can reasonably discount. This requirement is considerably more rigorous than just trying to match a few physical attributes of a single halo as these much more likely to agree well just by chance.

Not surprisingly the degree to which the positions agree over time depends strongly on the set of particles selected for comparison: we observe larger differences in the redshift zero relative positions of samples of particles chosen to be close to the halo centre than further out beyond the virial radius. For our sample, we will take all particles between 200 and 300 kpc h^{-1} of the potential centre of the redshift zero reference halo. This choice is somewhat arbitrary, and is ultimately made on aesthetic grounds: we find that our test measure, described later, shows a greater variation between the different sets of initial conditions than a sample chosen from within the halo itself. This variation makes for clearer figures. In fact, the conclusions we draw are insensitive to the sample choice provided the particles are taken from the high-resolution region.

We define the positions of the particles relative to the halo centre in each simulation, so the relative positions of particles between initial conditions are insensitive to translations of the haloes. This means the differences in the halo positions, as defined by their potential minima, are an independent measure.

The distribution of the relative particle displacements between the end states of two simulations has a very long tail to large separations: there are always some particles that end up on opposite sides of the halo. The majority of particles however typically have a much narrower spread. To avoid being strongly influenced by the tail of the distribution, which contributes little to the total mass density, we choose the median of the distribution of separations as the test measure and call it ΔR. We have checked that taking other percentiles such as 25 or 75 per cent makes no significant difference to the rankings of pairs of simulations. For convenience, we will measure ΔR in units of h^{-1} kpc.

To get some intuition as to how this measure behaves, we first test it between pairs of simulations that are extremely similar. It is well known that the end states of N-body simulations can be very sensitive to extremely small changes in their initial conditions (Miller 1964). We can see this effect using ΔR as a measure. Taking the reference set of initial conditions, we make a new set of initial conditions by copying them and modifying the velocities of each particle, which are represented as single precision floating point numbers, and randomly perturb each velocity up or down by the smallest amount possible. We use different random sequences to produce five sets of perturbed initial conditions. This perturbation introduces a fractional 'error' on the velocities of about one part in 10 million. It is inevitable at least with single precision velocities that any initial conditions will contain errors at least of this magnitude.

We then ran these five sets of simulations to redshift zero and looked at the median differences in the particle positions between the 10 pairs of initial conditions. The values of ΔR in h^{-1} kpc are shown below:

6.4

6.2 6.0

6.3 6.0 6.1

6.5 5.9 5.9 6.1.

The mean differences in ΔR between all the pairs are remarkably consistent. In size they are slightly greater than 4 per cent of R_{200}. Had we taken a sample of all particles within 300 h^{-1} kpc instead, then the differences would be about 18 h^{-1} kpc which is more than 10 per cent of the virial radius. These differences are largely random rather than systematic as the physical properties of the halo in these five different versions are extremely similar to each other and to the reference halo. The fractional rms variation in the quantities M_{200}, $V_{\rm disp}$ and $V_{\rm max}$ are 0.1, 0.15 and 0.2 per cent, respectively. These uncertainties give a measure of how accurately one can reasonably expect to determine these quantities even with extremely high quality initial conditions and a lower limit of what to expect for ΔR.

6.3 Testing cosmological initial conditions

Having established a benchmark for the size of ΔR for virtually identical initial conditions, we will now look at the differences in the final halo between sets of cosmological initial conditions. We expect the quality of the phase reconstruction to depend strongly on the size of the Fourier grids used. The larger the Fourier grid, the more information from Panphasia can be used to generate the phases. As described in the last section, all sets of initial conditions contain additional information from a field which is uncorrelated with Panphasia and that is introduced to restore isotropy to the initial conditions particularly at small scales. We will call this field the 'independent field'. We can investigate the influence of adding this independent field further by generating an ensemble of initial conditions that differ only in using a different realization of the independent field.

We can make cosmological initial conditions using the information provided by all eight of the Legendre block coefficients, or just the p_{000} block. This allows us to compare the relative merits of using either the S_8 or S_1 octree basis functions. From Figs 1 and 3, we expect to see significant differences and dependences on the Fourier grid size.

Table 3 shows all the cosmological initial conditions we use in this subsection. The top set of initial conditions in the table is the reference calculation described earlier. The 'Ref-alt' set is identical to the reference calculation except that it has a different realization of the independent field. These two sets of initial conditions were very expensive to set up so we have made do with just two realizations of the independent field. The Cosm-6144 and Cosm-3072 sets are analogous to the reference simulation but were set up using 6144^3- and 3072^3-sized Fourier grids instead. For each of these, we generate five further sets with different realizations of the independent field as indicated by '-alt' postfix to the names. In addition, there are two sets Cosm-6144-S_1 and Cosm-3072-S_1 which were made up using the S_1 octree basis functions with 6144^3 and 3072^3 Fourier grids, respectively. Finally in the table the five sets of initial

Table 3. Cosmological initial condition sets used in Fig. 5. The index is used as a label in that figure. See the main text for more details.

Index	Name	1D FFT size	S_1/S_8
1	Reference	12 288	8
2	Ref-alt	12 288	8
3	Cosm-6144	6144	8
4	Cosm-6144-alt1	6144	8
5	Cosm-6144-alt2	6144	8
6	Cosm-6144-alt3	6144	8
7	Cosm-6144-alt4	6144	8
8	Cosm-6144-alt5	6144	8
9	Cosm-3072	3072	8
10	Cosm-3072-alt1	3072	8
11	Cosm-3072-alt2	3072	8
12	Cosm-3072-alt3	3072	8
13	Cosm-3072-alt4	3072	8
14	Cosm-3072-alt5	3072	8
15	Cosm-3072-S_1	6144	1
16	Cosm-6144-S_1	3072	1
17	Ref-fp1	12 288	8
18	Ref-fp2	12 288	8
19	Ref-fp3	12 288	8
20	Ref-fp4	12 288	8
21	Ref-fp5	12 288	8

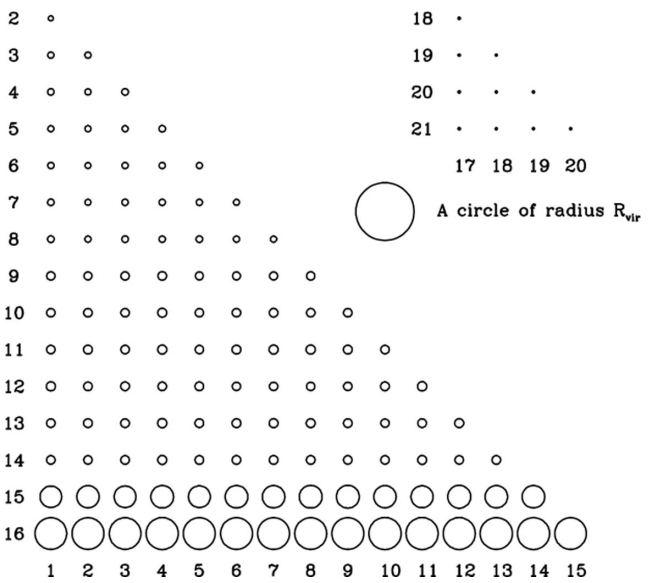


Figure 5. Each circle corresponds to a comparison made between two completed simulations started from the initial conditions listed in Table 3. The numbers in the rows and columns correspond to the index given in the table. The radius of each circle is proportional to the quantity ΔR defined in Section 6.2. The smaller the circle, the better is the agreement. A circle of radius $R_{\rm vir}$ is shown for scale.

conditions with a '-fp' postfix are those introduced in the previous subsection. These differ from the reference set by the smallest differences possible for single precision floating point representations of the particle velocities.

The large and small triangles made of circles in Fig. 5 show a set of comparisons between pairs of initial conditions listed in Table 3. The row and column numbers of each circle correspond to the indices given in the table and show which pair of simulations is being compared. The radius of each circle is proportional to the value of ΔR evaluated between the pair of haloes generated from the initial conditions. For comparison, the large isolated circle has a radius of 157 h^{-1} kpc, corresponding to R_{200} of the reference halo.

The observations from Fig. 5 are given as follows.

(i) The smallest circles are those corresponding to the comparisons between sets 17–20 and represent the smallest differences we might reasonably expect to see between pairs of initial conditions given the presence of single precision floating point errors.

(ii) The smallest circle in the main triangle is for the pair of simulations run from initial conditions made with the S_8 octree basis functions and the largest Fourier transform size: $12\,288^3$, as we would expect from the predictions of the linear power spectrum shown in Fig. 3. This circle has a radius of 13.5 h^{-1} kpc which is about a factor of 2 greater than what is potentially achievable at the floating point limit. As the separations are in three-dimensional space, the associated change in volume corresponds to a factor of 10 worse. The addition of the independent field does have a measurable effect on the accuracy of the phases even when using a $12\,288^3$ Fourier transform. However, as we will see later that the effect on the bulk halo properties such as virial mass and maximum circular velocity is sufficiently small as to be indiscernible in practice.

(iii) Below this, the circles making rows 3–8 are comparisons between pairs of initial conditions where one was made with a 6144^3 Fourier grid and the other a 6144^3 Fourier transform or greater. These 27 circles are all very similar in size ranging from 15.9 to 18.0 h^{-1} kpc with a mean of 16.9 h^{-1} kpc. In volume terms ΔR^3 is a factor of 2 larger when compared to the results for a $12\,288^3$ Fourier transform.

(iv) The circles in rows 9–14 consist of comparisons between pairs with one made using a 3072^3 Fourier grid and the other a 3072^3 Fourier grid or larger. Again these circles are remarkably similar in size and distinct from those above. They range in size from 21.5 to 24.9 h^{-1} kpc with an average of 23.2 h^{-1} kpc. $(\Delta R)^3$ is a factor of 5 larger than for the pair generated with a $12\,288^3$ Fourier transform.

(v) Finally, the bottom two rows consist of comparisons where one of the initial condition sets was made using the S_1 octree basis functions. A 6144^3 Fourier grid was used for the penultimate row, and a 3072^3 grid for the bottom row. The average ΔR value for a 6144^3 Fourier grid is 60 h^{-1} kpc, and for a 3072^3 grid it is 86 h^{-1} kpc. In terms of $(\Delta R)^3$, these values are about 90 and 260 times, respectively, larger than using the S_8 octree basis functions on a $12\,288^3$ Fourier transform.

The consistency of the ΔR measure between the ensembles of initial conditions made with 6144^3 and 3072^3 Fourier transforms suggests that the one measurement between the pair of simulations using a $12\,288^3$ Fourier transform would likely to be representative of an ensemble, had we been able to afford to make them.

The trends seen in Fig. 5 are consistent with what would be expected: using a larger Fourier grid leads to a better match to the reference simulation, and using the S_8 octree basis functions is much better than using the S_1 set. The ΔR measure shows these trends very clearly. There is no overlap in the sizes of the circles between the sets of rows described above. While the trends in ΔR have a clear physical interpretation, it is not obvious how these values are related to errors in reproducing simple measured bulk properties for a halo.

In Table 4, we compare the values of M_{200}, $V_{\rm disp}$ and $V_{\rm max}$ for the redshift zero halo from all of the runs listed in Table 3. Where there

Table 4. Physical parameters of the dark matter halo at redshift zero for a series of simulations running using initial conditions listed in Table 3. The column headings refer to initial conditions with indices in the table as follows: Reference 1; Ref-fp 17–21; $6144^3/S_8$ 3–8; $3072^3/S_8$ 9–14; $6144^3/S_1$ 16; $3072^3/S_1$ 15. In columns where there is more than one simulation, the numbers given are the average of the quantity for the sample together with the estimated rms about the average.

Quantity	Reference	Ref-fp	$6144^3/S_8$	$3072^3/S_8$	$6144^3/S_1$	$3072^3/S_1$
Number of simulations	1	5	6	6	1	1
M_{200} ($10^{10}\,h^{-1}\,\mathrm{M}_\odot$)	91.61	91.59 ± 0.09	91.52 ± 0.12	91.41 ± 0.12	90.08	86.16
$V_{\rm disp}$ (km s^{-1})	102.72	102.89 ± 0.17	102.80 ± 0.27	103.26 ± 0.18	102.28	99.72
$V_{\rm max}$ (km s^{-1})	181.71	181.83 ± 0.30	181.70 ± 0.37	182.23 ± 0.40	182.22	179.27

are several similar initial conditions, we compute a mean and rms about that mean. With only quite small samples the estimates of the rms themselves have a significant error.

We can see that the cosmological initial conditions made using S_8 and a 6144^3 Fourier transform have properties that are close to the reference calculation. The estimated rms values are comparable to or smaller than the differences between the means. For the results using the 3072^3 Fourier transform, the level of agreement is still very good, but the differences between the mean and the reference calculation are larger.

This situation for the two halo simulations run from initial conditions made using S_1 octree basis functions is much less satisfactory. The two haloes produced from these initial conditions differ from the reference halo by a much wider margin than any other halo. As expected, using a larger Fourier grid does improve the quality of the initial conditions. The result for a 6144^3 Fourier transform does match $V_{\rm max}$ well, but this is likely a coincidence as we know that the underlying particle distributions are significantly different.

The ΔR statistic is by construction insensitive to whether the halo potential centres agree or not. Comparing the absolute positions of the halo centres we find that all simulations run from initial conditions created using S_8 agree in the position of the potential minimum to better than $2\,h^{-1}$ kpc, while the two simulations run from initial conditions made using S_1 differ from each other and all other simulations by more than $10\,h^{-1}$ kpc.

The degree of agreement in the physical properties between the reference halo and the others is broadly consistent with the trends in ΔR although less clear cut. The cosmological initial conditions using the S_8 octree basis functions with a 6144^3 Fourier grid appear to be indistinguishable from the reference halo for the bulk halo properties M_{200}, $V_{\rm disp}$ and $V_{\rm max}$. The agreement with those made with a 3072^3 grid is extremely close and at the sub-per cent level. Using S_1 octree basis functions, however, leads to much poorer results with properties such as M_{200} differing by more than 4 per cent when a 3072^3 grid is used.

We conclude from these comparisons of the bulk properties of the halo that it is possible to generate high-quality cosmological initial conditions that accurately reconstruct the phase information given by Panphasia. The results obtained using the S_8 set of Legendre blocks are sufficiently good that it is not obvious that going to more complicated octree basis function expansions such as S_{27} would reproduce the bulk properties of the halo any more accurately. It is clear however that just using the S_1 octree basis functions gives poor quality initial conditions and is therefore not recommended.

6.4 Testing the resimulation method

Having established the quality of the initial conditions made using the cosmological method, we can now use them to test the quality of the resimulation initial conditions made using the method outlined in Section 5.3. The main goal of this subsection, and indeed the section, is to demonstrate that it is possible to make good quality resimulation initial conditions with Panphasian phases. To do this, we will simply compare the bulk properties of a resimulated version of a halo to the reference halo and show that they agree at the sub-per cent level. We will then use the more rigorous ΔR measure to check this conclusion.

Table 5. The properties of the resimulated halo and a comparison to the reference halo. The quantities and their units are explained in Table 2. The third column gives the difference between the resimulated halo and the reference halo. For the directional cosines, the angle between the spin directions is given.

Property	Value(s)	ΔReference
Centre	(43.1712, 49.9135, 2.2093)	(0.0020, 0.0035, 0.0023)
M_{200}	91.32	−0.29
$M_{\rm vir}$	109.96	−0.06
R_{200}	157.77	−0.18
$R_{\rm vir}$	213.43	0.04
$V_{\rm disp}$	102.89	0.17
$V_{\rm max}$	181.70	−0.01
λ'	0.0133	0.0000
($c1$, $c2$, $c3$)	(−0.216, −0.641, −0.736)	0.°75

There are quite a few choices that need to be made when setting up resimulation initial conditions. In this subsection, we detail these choices without justification. We explore in the next subsection how varying these choices affects the quality of the initial conditions.

For these resimulation initial conditions, we use four Fourier meshes centred around the high-resolution region with linear sizes of 1, 1/2, 1/4 and 1/8 of the periodic volume. We use a 768^3 Fourier transform for all meshes. The memory requirement of the IC_2LPT_GEN code for this grid size is about 10 Gb which means the code can comfortably fit on a modern supercomputer node. We will take the value of the 'X' parameter, introduced in Section 5.3, to be 4. We will use S_8 octree basis functions.

Fig. 4(b) shows an image of the resimulated halo. Clearly, it resembles the reference halo closely. Table 5 gives some of the bulk properties of this halo and shows how it differs from the reference halo. The differences are remarkably small – at the sub-per cent level. We find similar levels of agreement for resimulations of other haloes using the same methods so these numbers can be taken as typical. We conclude that is possible to make high-quality resimulation initial conditions from Panphasia.

This conclusion is supported by the measured value of ΔR between the reference halo and the resimulated halo. The value is $22.8\,h^{-1}$ kpc, which is slightly better than the difference in the mean between the reference halo and the haloes generated from

cosmological initial conditions made with a 3072^3 Fourier transform. By this measure the resimulation initial conditions are found to be of similar quality as can be created using single very large Fourier transforms. In effect, we can say that the resimulation method is able to produce initial conditions of comparable quality to what can be achieved using the Fourier method developed in the 1980s to model cosmological volumes, but at a considerably lower computational cost.

Encouraging as these results are, we can see from the value of ΔR that it may be possible to make better initial conditions than we have achieved. Compared to the difference in ΔR between the two sets of cosmological initial conditions made with a $12\,288^3$ Fourier transform, the difference in $(\Delta R)^3$ is a factor of 4. This is a reasonable comparison as the mesh spacing of the innermost grid of the resimulation initial conditions is the same as for the cosmological initial conditions made with a $12\,288^3$ Fourier transform so the same information from Panphasia is used for the inner region.

6.5 Sensitivity to parameters

In this subsection we show how the choice of the Fourier mesh size, the 'X' parameter (defined by rule 4 of Section 5.3) and the choice of S_8 or S_1 octree basis functions affect the accuracy with which the phase information can be reconstructed in resimulation initial conditions. We expect the results to improve with the size of the Fourier grid. This is both because more Panphasia information is used and because each given octree basis function is sampled more finely by the Fourier grid. We can maximize the sampling of the octree basis functions by applying rule 3 of Section 5.3, which states that we should place octree functions on the finest possible grid. However, we introduced a fourth rule, which forbids large octree basis functions being placed close to the boundaries of any but the outermost grid. The reasoning for this was that placing large octree basis functions near the boundaries of the inner grids would lead to larger edge effects. We expect that for very large values of X this rule will tend to place octree basis functions on coarser grids, and will lead to poorer reconstruction of the phases because of poor sampling of the octree functions. For very small values of X, boundary effects may also have a negative effect. Finally, we expect better results using the S_8 octree basis functions compared to the S_1 set as we saw for cosmological initial conditions.

We have made sets of initial conditions with a wide range of combinations of Fourier grid sizes, X values and types of octree basis function: the Fourier grid sizes used are 384^3, 768^3 and 1536^3; the values of X are 2, 3, 4, 6, 8, 12, 16, 24, 32, 48 and 64; and octree basis functions are S_8 and S_1. We then run each set of initial conditions to redshift zero and have computed ΔR with respect to the reference halo.

The results of those comparisons are shown in Fig. 6. As expected, we see a clear trend with Fourier grid size, and between using the S_8 and S_1 basis functions. The best result using S_1 and a 1536^3 Fourier transform is still poorer than the best using S_8 and a 384^3 grid. The former initial conditions are also almost two orders of magnitude more expensive to generate. This confirms for resimulation initial conditions the conclusion we had already reached for cosmological simulations that the S_1 basis functions are unsuitable for accurate work.

Concentrating on the S_8 results, we see in Fig. 6 that larger values of X are clearly disfavoured for the 384^3 and 768^3 grids. There is almost no trend for a 1536^3 grid. For this grid size, the value of ΔR does show an increase for $X = 128$ and 256 (not shown) where the values of ΔR are 26.9 and 37.3 h^{-1} kpc, respectively. Choosing a maximum value of X that depends on the grid size gives an upper limit to the X parameter. Using a value $X < M/24$, where M is the Fourier grid choice, appears a safe choice, with larger values clearly disfavoured.

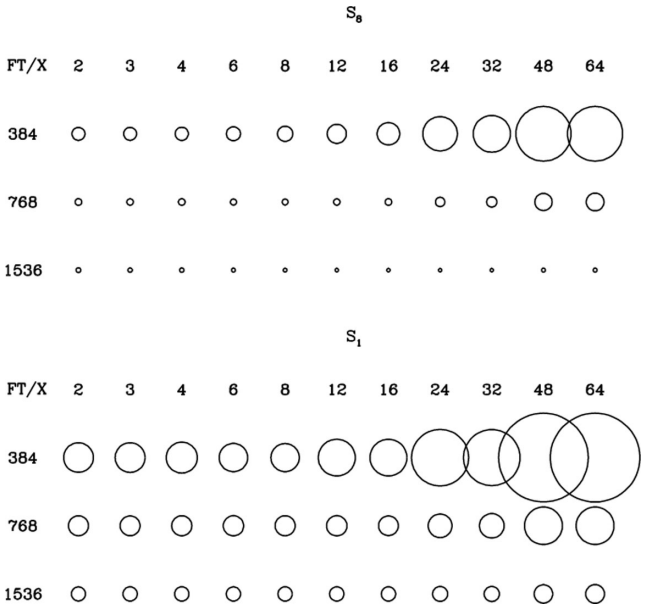


Figure 6. Each circle corresponds to a comparison between the reference halo and a halo produced by running a set of resimulation initial conditions, set up using the parameter values shown labelling the rows and columns. The rows are labelled by the one-dimensional size of the Fourier transform used and the columns by the value of the X parameter. The upper set of circles use the S_8 octree basis functions, the lower set the S_1 ones. The smallest circle corresponding to FT = 1536, $X = 12$, S_8 has a radius of 20.0 h^{-1} kpc. By contrast, the largest circle has a radius of 260 h^{-1} kpc.

While large values of the X parameter give poor results, the range of X explored does not show any convincing evidence for boundary effects being important at low values of X. It is not true however that the lowest values of ΔR occur for the lowest values of X; it is just that there is no clearly distinct minimum in ΔR values. We conclude from these tests that the quality of the resimulation initial conditions is not that sensitive to precisely in which grid the octree basis functions are placed, above a minimum sampling. The choice of $4 \leq X \leq 12$ appears to work well for all grid sizes we have tested.

The smallest circle in the diagram corresponds to $X = 12$ and a 1536^3 Fourier transform and has a value of ΔR of 20.0 kpc h^{-1}. This value is intermediate between that found for the 3072^3 and 6144^3 cosmological initial conditions. The run time to make the resimulation initial conditions with a 1536^3 Fourier grid is however longer than the N-body simulation takes to go from redshift 63 to redshift zero. This seems rather excessive for a real application. For this reason, we chose to use a 768^3 Fourier transform for the last subsection to demonstrate the quality of the resimulation initial conditions. With this grid size, the lowest value of ΔR is 22.8 h^{-1} kpc for $X = 4$. These particular 2LPT initial conditions take about 20 min to be made on a single 16-core node. Zeldovich initial conditions would be made considerably quicker. The N-body simulation on the same node takes about 2 h. For more sophisticated simulations of structure formation modelling hydrodynamics and complex subgrid models, the run time may be much longer than

Table 6. The phases of several recent Virgo Consortium simulations. The text phase descriptor gives the location of the phase information in Panphasia. See Section 7.1 for details. The MW7 simulation has been added to the Millennium data base (Guo et al. 2013).

Simulation	Reference	Phase descriptor
DOVE	Not yet published	[Panph1,L16,(31250,23438,39063),S12,CH1292987594,DOVE]
MW7	Guo et al. (2013)	[Panph1,L11,(200,400,800),S3,CH439266778,MW7]
MXXL	Angulo et al. (2012)	[Panph1,L10,(800,224,576),S9,CH1564365824,MXXL]

that for the pure N-body simulation. In such cases, the fractional costs of making the initial conditions become relatively small.

7 PUBLISHING PHASE INFORMATION

All that is needed to specify the phase information of initial conditions generated from Panphasia is the spatial location of the phases within the volume. To use this information to resimulate a region of interest within a simulation requires specifying the position and dimensions of the region at high redshift.

In Section 7.1, we define a convention for specifying phases taken from Panphasia and using this convention publish the phases of three cosmological volumes run by the Virgo Consortium. In Section 7.2, we give the locations and sizes of a region within each of these volumes out of which forms a dark matter halo at redshift zero. We show images and give the positions and a few properties of these haloes for reference.

7.1 How to publish Panphasia phases

For the purposes of designing a convention for publishing phase information, we will assume that the phase information for all simulations is defined by specifying either a cube or a cuboid within Panphasia made of complete octree cells. The location of a cube within an octree requires five integers: one integer to define the smallest level in the octree that is possible; three integers to specify the location of the corner cell closest to the origin, using the Cartesian coordinates described in Section 2.2 to label the cells; and one integer to give the side length of the cube in units of the octree cell at the given level. For a general cuboid, or one with two different side lengths, we will require three side lengths to be given.

Taking these integers we define two text phase descriptors incorporating these numbers: one for cubic regions and one for a general cuboid. Because making an error in the value of any of the integers in a descriptor would change the phase information, we include an additional check number in the descriptor. To be useful this check integer must depend on all of the integers that define the phase. While having an error check can avoid some human errors, it is no safeguard against simply using the wrong descriptor. It is desirable for the descriptor to also include a human readable name that can be readily associated with a particular simulation volume.

The check number we have selected combines the random number states associated with the three corner cells adjacent to the corner cell nearest to the origin and the name of the phase descriptor. The full details are given in Appendix B.

For a cubic region, we define a plain text phase descriptor:

[Panph1, L#$_0$, (#$_1$, #$_2$, #$_3$), S#$_4$, CH#$_5$, STRING],

where each symbol # represents an integer: #$_0$ is the octree level, (#$_1$, #$_2$, #$_3$) are the Cartesian coordinates of the corner cell nearest to the origin, #$_4$ is the side length and #$_5$ is the check number. For a cuboidal region, we define a second phase descriptor:

[Panph1, L#$_0$, (#$_1$, #$_2$, #$_3$), D(#$_4$, #$_5$, #$_6$), CH#$_7$, STRING],

where the three side lengths are given by (#$_4$, #$_5$, #$_6$) for each Cartesian direction and #$_7$ is again the check number. Finally, STRING is a text string, again without spaces, naming the particular realization of the phases. This should be distinctive as this is the best protection against using the wrong descriptor by accident.

There are no spaces within the phase descriptor and the type of brackets punctuation, and cases of the letters should be observed. The string 'Panph1' is intended to help identify the descriptor and to make it possible to make a text search for Panphasia descriptors. The '1' allows for the possibility of extending or adapting the format in the future.

A code to randomly generate phase descriptors, including the check digit, is included with the public release of Panphasia.

Table 6 gives the phase descriptors for three cosmological simulations run by the Virgo Consortium including the DOVE simulation from which we resimulated the reference halo. In the next subsection we give examples of a halo that can be resimulated from each of these volumes.

7.2 Example haloes for resimulation

For anyone wanting to implement Panphasian phases in a new code, it is desirable to have some test cases to check that the code is working correctly. In Table 7, we give the locations and sizes of a single sphere within each of the DOVE, MW7 and MXXL volumes at high redshift. A resimulation of these spheres results at redshift zero in the formation of a prominent dark matter halo selected from these cosmological volumes. Images of these haloes at redshift zero, projected on to the x_1-x_2 plane, are shown in Figs 7–9. The DOVE and MW7 volumes have the same cosmological parameters, given in Table 1. The MXXL parameters are different and are given in Angulo et al. (2012). We use the same coordinate system to describe these locations as for Panphasia: the coordinates are non-negative and the origin marks one corner of the volume.

Table 7. Locations of a sphere within each volume from which a sizeable dark matter halo forms. The first three digits within the round brackets mark the centre of the sphere, while the fourth gives the radius of the sphere. The coordinates of the simulation volume include the origin and are non-negative.

Simulation	Position and size of halo Lagrangian region (h^{-1} Mpc)
DOVE	[(41.11,48.80,3.00),4.5]
MW7	[(307.46, 52.51,434.33),46]
MXXL	[(1806.8,1207.5,1617.9),35]

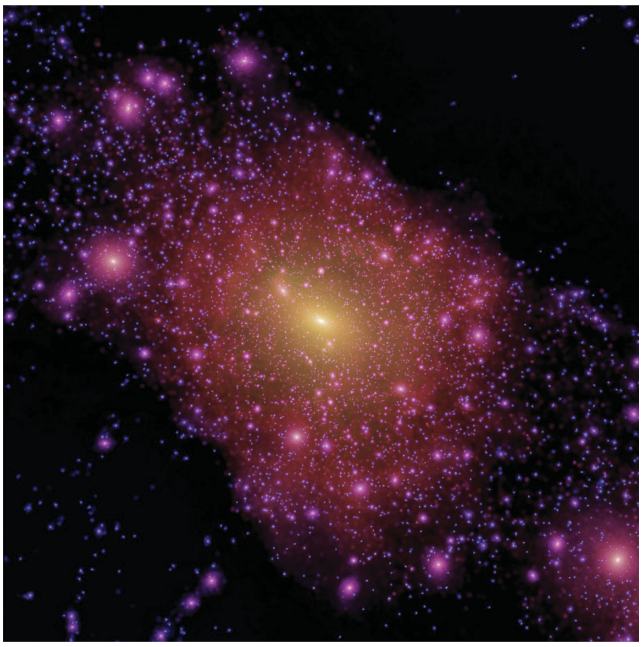


Figure 7. High-resolution version of the DOVE reference halo used in this paper. The projection is the same as in Fig. 4.

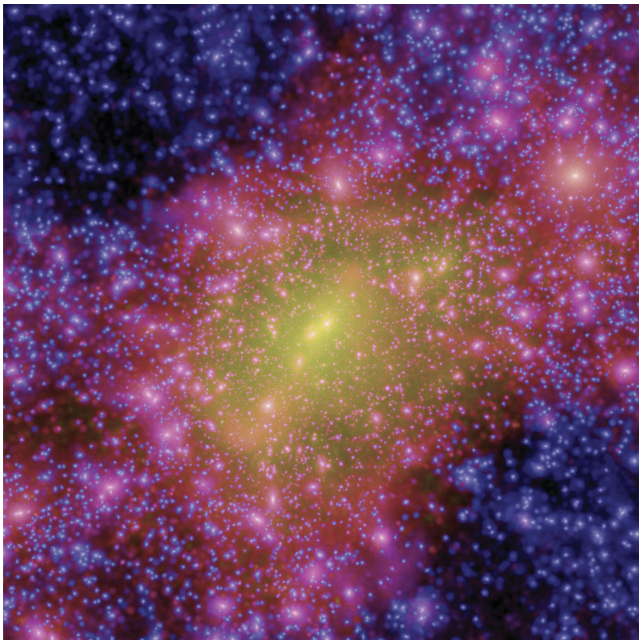


Figure 8. High-resolution version of the MW7 reference halo at redshift zero. The projection is in the x_1-x_2 plane and the side length of the image is $10\,h^{-1}$ Mpc. This is the most massive cluster in the MW7 volume at redshift zero.

The redshift zero properties of these haloes, as determined by a resimulation using the methods described in this paper, are given in Table 8. The halo in the DOVE volume is a resimulation of the same volume as was used for the reference halo in Section 6, but with a factor of about 30 more particles. The properties of this halo are very similar to those obtained in the reference calculation. As this resimulation has additional small-scale power, it does not follow that the value of M_{200} should be precisely reproduced. The difference we observe between the two versions of the halo at different resolutions

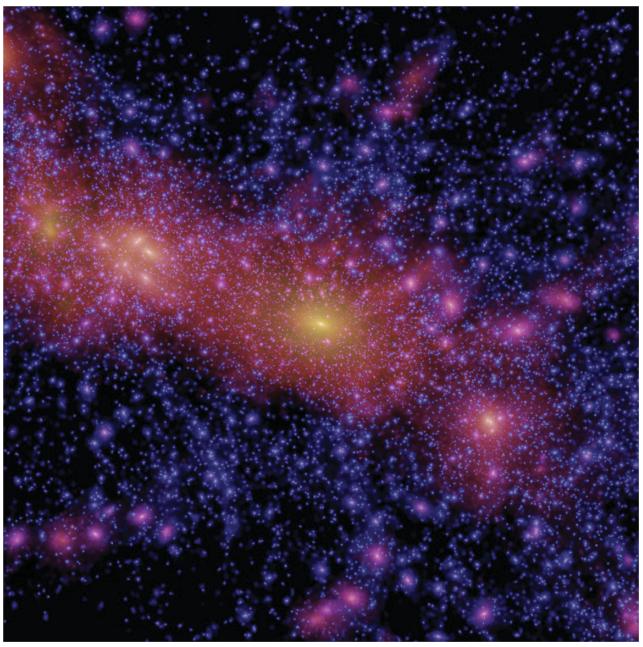


Figure 9. High-resolution resimulation of a reference halo in the MXXL simulation. The projection at redshift zero is in the x_1-x_2 plane, and the image measures $15\,h^{-1}$ Mpc on a side.

Table 8. Location and a few properties of reference haloes at redshift zero. The position (x_1, x_2, x_3) is the location of the particle with the lowest potential as determined by SUBFIND. The quantity m_p is the mass of the particles in the high-resolution region of the resimulation. The other quantities are defined in the caption of Table 2.

Halo property	Unit	DOVE	MW7	MXXL
x_1	h^{-1} Mpc	43.174	303.06	1802.15
x_2	h^{-1} Mpc	49.918	52.90	1205.52
x_3	h^{-1} Mpc	2.203	423.55	1614.56
m_p	$10^6\,h^{-1}\,\mathrm{M}_\odot$	0.115	133.54	51.65
M_{200}	$10^{10}\,h^{-1}\,\mathrm{M}_\odot$	90.81	117 756	42 433
R_{200}	h^{-1} kpc	157.47	1717	1222
$V_{\rm disp}$	km s^{-1}	104.8	1054	737
$V_{\rm max}$	km s^{-1}	182.1	1673	1305

is similar in size to that seen in Springel et al. (2008) between the Aq-A-5 and Aq-A-4 resimulations. This particular halo has also been simulated at very similar numerical resolution to the example given in this subsection with initial conditions made using a single 6144^3 Fourier grid. The properties of the two higher resolution versions of this halo are a very good match to each other with smaller difference than seen between versions of the halo simulated with very different particle numbers.

The haloes in the MW7 and MXXL volumes are both in the cluster mass range. Both these haloes are in the process of formation and are far from equilibrium. Each is resolved with about 8 million particles within R_{200}.

8 OVERVIEW OF THE PUBLIC CODE TO GENERATE PANPHASIA

In this section we give a brief overview of the code to generate Panphasia. The full details are given in the companion paper

(Jenkins & Booth 2013), which being on the arXiv may be updated after this paper is published. The following therefore provides only a very general overview.

From the point of view of adding Panphasia phases to an existing code for making initial conditions, there are just two subroutines that need to be called directly. The first is an initialization routine which takes the Panphasia phase descriptor. The second is an evaluation routine that returns values of the field itself for a location chosen by the user.

The routines are serial codes. They take very little memory however so there is no significant cost to a parallel code in having multiple instances of the code – with one belonging to each mpi process for example. The code is therefore easy to use in parallel applications: different parts of the white noise field can be computed completely independently. The main parallel programming needed to incorporate these routines is to ensure that the white noise field is assigned to the correct locations as dictated by the way the relevant grids are distributed in parallel by the application code. As discussed in Jenkins & Booth (2013) the speed of the code does depend significantly on the precise ordering of successive accesses to octree cells and it is therefore important to consider the effects of the access pattern when adding Panphasia to a parallel code.

For making cosmological conditions, these two subroutines are especially simple. The initialization routine needs the Panphasian phase descriptor and the size of the grid that will be used to make the initial conditions. The latter is needed to decide which level of the octree to sample the region of Panphasia selected by the descriptor. The evaluation routine just needs to be called with three integer Cartesian grid coordinates, and returns nine Gaussian pseudo-random numbers – all drawn from a distribution with unit mean and unit variance. The first eight of these are proportional to the expansion coefficients of Panphasia expanded in the eight Legendre block functions, while the ninth independent value can be used to construct a field that is independent of Panphasia.

For resimulation initial conditions, a more general initialization routine is provided which requires a refinement within the cosmological volume to be specified. The evaluation routine in this case returns the same nine values, but now as a function of three integer coordinates that are defined relative to the refinement origin. This function operates in the same way even if the refinement is wrapped by periodic boundary conditions across a simulation coordinate boundary. In fact, the halo we resimulated from the DOVE volume is located close to a coordinate boundary so this feature is tested in this paper.

The refinement evaluation routine also takes values for a minimum and maximum octree level. The values of the eight Legendre blocks returned are calculated using only the octree functions over the range of octree levels specified. In this way, the user can decide how to place the octree basis functions. The method for placing the octree basis functions used by IC_2LPT_GEN for setting up the resimulations in this paper is determined by a subroutine which gives suggested minimum and maximum values for the octree levels as a function of position in the grid. This subroutine is included with the public code.

As well as the routines to evaluate Panphasia, we also provide a routine to choose a random region from Panphasia and generate a phase descriptor. For this purpose, it is assumed that Panphasia itself has a physical size. The user must specify both the physical size of the cosmological volume and the required dimension of volume (assumed to be a cube) measured in grid cells. The code uses the user supplied information together with the Unix timestamp to generate a descriptor.

It is assumed that the root cell of Panphasia measures 25 000 h^{-1} Gpc on a side. This gives a volume which is about 10 billion times the current Hubble volume for ΛCDM. At the same time, the mass associated with the smallest defined octree cells at level 50 of the octree has a corresponding mass of about $10^{-12}\ h^{-1}\ \mathrm{M}_{\odot}$, which is below the cut-off scale for a WIMP dark matter candidate which is estimated to be around $10^{-7}\ h^{-1}\ \mathrm{M}_{\odot}$ (Hofmann et al. 2001).

Finally, as an example we add Panphasia to a serial public initial conditions code described in Crocce, Pueblas & Scoccimarro (2006).[3]

9 SUMMARY AND DISCUSSION

In this paper, we describe a new way for setting the phase information for Gaussian initial conditions for cosmological simulations and resimulations of structure formation. This work builds upon the idea of using a real-space white noise field to define the phase information – a method put forward by Salmon (1996) and implemented by a number of authors including Pen (1997), Bertschinger (2001) and Hahn & Abel (2011). We have developed a way of defining Gaussian white noise fields in terms of a basis function expansion using purpose designed orthogonal basis function sets that have a hierarchical structure based around an octree. Vast realizations of Gaussian white noise fields can be easily created by assigning expansion coefficients, which are created by a pseudo-random number generator, systematically to the space of basis functions. Using a pseudo-random number generator that allows rapid access to any part of the sequence results in the creation of what is effectively an objective Gaussian white noise field sampled over a very wide range of spatial scales, any part of which is readily accessible.

We have chosen a particular set of octree basis functions that we find on the basis of tests to be most suitable for making cosmological initial conditions. This choice is a compromise between the accuracy with which the phases of large-scale modes can be reproduced from a finite number of octree functions and the computational cost of evaluating the white noise field. We have found that the simplest choice with seven distinct functional forms for the octree basis functions is unsuitable for accurate simulation work. We show through resimulation tests that a choice based on 56 functional forms (that can however be expressed in terms of eight more primitive functions) is sufficiently accurate as judged against the published results of state-of-the-art resimulations of dark matter haloes.

We have created a particular realization of a Gaussian white noise field called Panphasia. This uses our preferred octree basis functions, with expansion coefficients derived from a commonly used pseudo-random number generator that passes very strong random number tests. We use almost the entire period of the generator to create a realization with 50 octree levels which is able to define phase information over 15 orders of magnitude in linear scale. We make Panphasia public by publishing in a companion paper, Jenkins & Booth (2013), a code to compute Panphasia. Small subregions of this larger field are suitable for setting the phases for cosmological simulations. The phases for these simulations can themselves be published by pointing to the location in Panphasia, from which the phase information was taken.

[3] We thank Martin Crocce for permission to include this modified code with our software.

To help with this we have defined a convention for publishing phase information for cosmological simulations set up using Panphasia. We have published the phases of three cosmological volumes run by the Virgo Consortium. As a guide for anyone wanting to add Panphasian phases to an initial conditions code, we have given the locations and properties of three dark matter haloes that can be resimulated within each of these cosmological volumes.

In order to demonstrate that it is possible to create high-quality resimulation initial conditions using phases from Panphasia, we have developed a method using Fourier transforms to do the numerical convolutions of the white noise field required to make initial conditions with a given CDM linear power spectrum. We are able to show that these methods work well by essentially making the same initial conditions using two different methods: as cosmological initial conditions using very large Fourier transforms – essentially the Fourier method in use since the 1980s – and by the new resimulation method described here. We find by looking at the properties of a dark matter halo at redshift zero that it is possible to recover the final positions of the particles to similar accuracy when using the resimulation method or the Fourier method. Similarly a number of halo properties are reproduced consistently between the two methods to sub-per cent accuracies. At the same time, these tests do show that there is some room for improvement in the resimulation initial conditions. It may well be that the very accurate multigrid methods developed by Hahn & Abel (2011) for the MUSIC code can be applied successfully to Panphasia and yield more accurate resimulation initial conditions than the methods described in this paper. This paper provides a guide on how to test the quality of resimulation initial conditions made using Panphasian phases. Any new implementation can be tested on the reference halo studied in Section 6 and so can be compared directly with the implementation applied in this paper to the IC_2LPT_GEN code.

While it is important that it is possible to make accurate resimulation initial conditions from Panphasia, this is not necessarily its most important feature. The fact that using Panphasia allows the phase information to be published by giving a short phase descriptor has potential benefits for all those involved in simulations of cosmological structure formation from Gaussian initial conditions. Using Panphasia provides a convenient way to keep track of how simulations were set up, and makes it possible for others to reproduce and check published simulation results, and to exploit existing simulations. It should also make it easier to apply very different numerical techniques to standard problems, for example in the field of galaxy formation, by using Panphasia as a convenient way to define the initial conditions.

These wider benefits will only accrue if Panphasia is commonly used. This requires two developments. First, Panphasia would need to be used when setting up large cosmological volumes – particularly where these simulations or products derived from them are made publicly available. Currently there are no simulations using Panphasian phases on the MultiDark data base (Riebe et al. 2011), and just one on the Millennium data base (Lemson & Virgo Consortium 2006). However, the Virgo Consortium is now using Panphasia, so this situation will improve in the future. Secondly, Panphasia needs to be added to existing initial conditions codes. It is relatively easy to add to codes that make cosmological initial conditions, but it will require some effort from a relatively small set of people to make Panphasian phases available in all existing resimulation codes.

If both these developments can be achieved, then anyone could use these data bases to select samples of objects for resimulation. An alternative to this however, also requiring investment of effort, would be for those providing the data bases to provide a service that serves resimulation initial conditions in the appropriate format to the users.

Finally, while this paper and its companion, Jenkins & Booth (2013), are intended to act as a self-contained guide on how to add Panphasia to other initial conditions codes, the author would be more than happy to provide help and advice to anyone interested in adding Panphasian phases to their codes.

ACKNOWLEDGEMENTS

I would particularly like to thank Stephen Booth of the Edinburgh Parallel Computing Centre for advice on pseudo-random number generators and kindly giving me his implementation of the MRGK5-93 pseudo-random number generator. Thanks to Ben Lowing, Shaun Cole, Carlos Frenk, Michael Schneider, Volker Springel and Simon White for comments on drafts or parts of drafts of this paper. Thanks also to Mark Lovell who made the striking colour images. This work was supported by the STFC rolling grant ST/F002289/1 and made use of the DiRAC I and II facilities at Durham, which are jointly funded by STFC, the Large Facilities Capital Fund of BIS and Durham University.

REFERENCES

Angulo R. E., Springel V., White S. D. M., Cole S., Jenkins A., Baugh C. M., Frenk C. S., 2012, MNRAS, 425, 2722
Bertschinger E., 2001, ApJS, 137, 1
Box G., Muller M., 1958, Ann. Math. Stat., 29, 610
Boylan-Kolchin M., Springel V., White S. D. M., Jenkins A., Lemson G., 2009, MNRAS, 398, 1150
Bullock J. S., Dekel A., Kolatt T. S., Kravtsov A. V., Klypin A. A., Porciani C., Primack J. R., 2001, ApJ, 555, 240
Crocce M., Pueblas S., Scoccimarro R., 2006, MNRAS, 373, 369
Efstathiou G., Davis M., White S. D. M., Frenk C. S., 1985, ApJS, 57, 241
Gao L., White S. D. M., Jenkins A., Frenk C. S., Springel V., 2005, MNRAS, 363, 379
Gao L., Navarro J. F., Frenk C. S., Jenkins A., Springel V., White S. D. M., 2012, MNRAS, 425, 2169
Guo Q., White S., Angulo R. E., Henriques B., Lemson G., Boylan-Kolchin M., Thomas P., Short C., 2013, MNRAS, 428, 1351
Hahn O., Abel T., 2011, MNRAS, 415, 2101
Hofmann S., Schwarz D. J., Stöcker H., 2001, Phys. Rev. D, 64, 083507
Jenkins A., 2010, MNRAS, 403, 1859
Jenkins A., Booth S., 2013, preprint (arXiv:1306.5771)
Komatsu E. et al., 2011, ApJS, 192, 18
L'Ecuyer P., Simard R., 2007, ACM Trans. Math. Softw., 33, 22
L'Ecuyer P., Blouin F., Couture R., 1993, ACM Trans. Model. Comput. Simul., 3, 87
Lemson G., Virgo Consortium t., 2006, arXiv:e-prints
Miller R. H., 1964, ApJ, 140, 250
Navarro J. F., Frenk C. S., White S. D. M., 1995, MNRAS, 275, 56
Pen U.-L., 1997, ApJ, 490, L127
Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P., 1992, in Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P., eds, Numerical Recipes in FORTRAN: The Art of Scientific Computing. Cambridge Univ. Press, Cambridge
Riebe K. et al., 2011, preprint (arXiv:1109.0003)
Salmon J., 1996, ApJ, 460, 59
Seljak U., Zaldarriaga M., 1996, ApJ, 469, 437
Springel V., 2005, MNRAS, 364, 1105
Springel V., White S. D. M., Tormen G., Kauffmann G., 2001, MNRAS, 328, 726
Springel V. et al., 2008, MNRAS, 391, 1685
Stadel J., Potter D., Moore B., Diemand J., Madau P., Zemp M., Kuhlen M., Quilis V., 2009, MNRAS, 398, 21

APPENDIX A: THE PANPHASIA OCTREE BASIS FUNCTIONS

In this appendix, we define the octree basis functions for Panphasia, in terms of the S_8 set of Legendre block functions, which themselves are defined by equation (13).

Before we can define the octree basis functions themselves, we first define, on the left-hand side below, a set of 64 functions each of which is some linear combination of Legendre block functions determined by the appropriate matrix equation on the right-hand side. The eight functions, for example P_{000}, heading each column on the left-hand side have by construction the functional forms of the Legendre blocks one level shallower in the octree. The 56 functions below these eight functions will be used below to define the S_8 octree basis functions themselves,

$$\begin{vmatrix} P_{000} \\ Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \\ Q_7 \end{vmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix} \begin{vmatrix} p_{000} \\ p_{001} \\ p_{010} \\ p_{011} \\ p_{100} \\ p_{101} \\ p_{110} \\ p_{111} \end{vmatrix}$$

Key:

$$\begin{vmatrix} P_{100} & P_{010} & P_{001} \\ Q_8 & Q_{15} & Q_{22} \\ Q_9 & Q_{16} & Q_{23} \\ Q_{11}{}^4 & Q_{17} & Q_{24} \\ Q_{10} & Q_{18} & Q_{25} \\ Q_{12} & Q_{19} & Q_{26} \\ Q_{13} & Q_{20} & Q_{27} \\ Q_{14} & Q_{21} & Q_{28} \end{vmatrix} = \begin{pmatrix} a_1 & a_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ -a_2 & a_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{vmatrix} p_{000} & p_{000} & p_{000} \\ p_{100} & p_{010} & p_{001} \\ p_{001} & p_{001} & p_{010} \\ p_{011} & p_{011} & p_{011} \\ p_{010} & p_{100} & p_{100} \\ p_{101} & p_{101} & p_{101} \\ p_{110} & p_{110} & p_{110} \\ p_{111} & p_{111} & p_{111} \end{vmatrix}$$

$$a_1 = \frac{\sqrt{3}}{2}$$

$$a_2 = \frac{1}{2}$$

$$\begin{vmatrix} P_{110} & P_{011} & P_{101} \\ Q_{29} & Q_{36} & Q_{43} \\ Q_{30} & Q_{37} & Q_{44} \\ Q_{31} & Q_{38} & Q_{45} \\ Q_{32} & Q_{39} & Q_{46} \\ Q_{33} & Q_{40} & Q_{47} \\ Q_{34} & Q_{41} & Q_{48} \\ Q_{35} & Q_{42} & Q_{49} \end{vmatrix} = \begin{pmatrix} b_1 & b_2 & b_2 & b_3 & 0 & 0 & 0 & 0 \\ -b_2 & -b_3 & b_1 & b_2 & 0 & 0 & 0 & 0 \\ b_3 & -b_2 & b_2 & -b_1 & 0 & 0 & 0 & 0 \\ -b_2 & b_1 & b_3 & -b_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix} \begin{vmatrix} p_{000} & p_{000} & p_{000} \\ p_{010} & p_{001} & p_{001} \\ p_{100} & p_{010} & p_{100} \\ p_{110} & p_{011} & p_{101} \\ p_{001} & p_{100} & p_{010} \\ p_{011} & p_{101} & p_{011} \\ p_{101} & p_{110} & p_{110} \\ p_{111} & p_{111} & p_{111} \end{vmatrix}$$

$$b_1 = \frac{3}{4}$$

$$b_2 = \frac{\sqrt{3}}{4}$$

$$b_3 = \frac{1}{4}$$

$$\begin{vmatrix} P_{111} \\ Q_{50} \\ Q_{51} \\ Q_{52} \\ Q_{53} \\ Q_{54} \\ Q_{55} \\ Q_{56} \end{vmatrix} = \begin{pmatrix} c_1 & c_2 & c_2 & c_3 & c_2 & c_3 & c_3 & c_4 \\ -c_2 & c_1 & c_2 & -c_3 & -c_2 & c_3 & c_4 & -c_3 \\ -c_2 & -c_2 & c_1 & -c_3 & c_2 & -c_4 & c_3 & c_3 \\ -c_2 & c_2 & -c_2 & c_4 & c_1 & -c_3 & c_3 & -c_3 \\ c_3 & -c_3 & -c_3 & -c_1 & c_4 & c_2 & c_2 & -c_2 \\ c_3 & c_3 & -c_4 & -c_2 & -c_3 & -c_1 & c_2 & c_2 \\ c_3 & c_4 & c_3 & -c_2 & c_3 & -c_2 & -c_1 & -c_2 \\ -c_4 & c_3 & -c_3 & -c_2 & c_3 & c_2 & -c_2 & c_1 \end{pmatrix} \begin{vmatrix} p_{000} \\ p_{001} \\ p_{010} \\ p_{011} \\ p_{100} \\ p_{101} \\ p_{110} \\ p_{111} \end{vmatrix}$$

$$c_1 = \sqrt{\frac{27}{64}}$$

$$c_2 = \frac{3}{8}$$

$$c_3 = \sqrt{\frac{3}{64}}$$

$$c_4 = \frac{1}{8}.$$

[4] It is just a feature of Panphasia that the ordering of Q_{10} and Q_{11} is reverse from what might have been expected.

The eight Legendre blocks each have distinct symmetries with respect to a reflection about the three principal coordinate planes about the cell centre. So for example p_{000}, which is a constant, has even parity with respect to all three reflections while p_{111} has odd parity for all three reflections. Before we can define the octree functions, we first define the following function, which is antisymmetric about the origin:

$$A(u) = \begin{cases} 1 & \text{if } 0 \leq u < 1; \\ -1 & \text{if } -1 < u < 0; \\ 0 & \text{otherwise.} \end{cases} \tag{A1}$$

Creating a product of three of these functions each acting on one of the Cartesian coordinates, we arrive at a useful function $A^i(x_1)A^j(x_2)A^k(x_3)$, which has the same parities about the origin as the corresponding Legendre block, $p_{ijk}(\boldsymbol{x})$, where $\boldsymbol{x} \equiv (x_1, x_2, x_3)$.

As stated earlier, the functions, P_{ijk}, heading each of the eight columns on the left-hand side of the matrix equation are linear combinations of Legendre block functions, and are simply related by construction to Legendre block functions of twice their linear scale by the following equation:

$$p_{ijk}(\boldsymbol{x}) = P_{ijk}\left(2|x_1| - \frac{1}{2}, 2|x_2| - \frac{1}{2}, 2|x_3| - \frac{1}{2}\right) A^i(x_1)A^j(x_2)A^k(x_3). \tag{A2}$$

Similarly taking the remaining seven functions in each of these columns, we can now define the functional forms of the octree basis functions themselves in an analogous way:

$$q_n(\boldsymbol{x}) = Q_n\left(2|x_1| - \frac{1}{2}, 2|x_2| - \frac{1}{2}, 2|x_3| - \frac{1}{2}\right) A^i(x_1)A^j(x_2)A^k(x_3), \tag{A3}$$

where n is an integer in the inclusive range $[7(i + 2j + 4k) + 1, 7(i + 2j + 4k) + 7]$, and i, j and k each take the values of zero or one.

Combining all eight columns yields 56 functions $q_n(\boldsymbol{x})$, $n = 1, 56$ which gives the functional form for the Panphasia octree basis functions, plus the eight Legendre block functions. These 64 functions when placed in any given octree cell are mutually orthogonal. This can be seen as follows. Functions drawn from different columns have different parities, and are therefore orthogonal. Within a given column the orthogonality can be verified by inspection of the orthogonality between pairs of rows of the square matrices, combined with the knowledge that the Legendre block functions in the columns on the right-hand side are themselves mutually orthogonal.

Because the 56 octree basis functions are orthogonal not only to each other, but also to the 8 Legendre block functions, and the octree basis functions are built of (smaller) Legendre blocks, it follows that any two different octree basis functions, placed in any two octree cells, are necessarily orthogonal, whether the octree cells overlap or not.

Using the fact that the octree basis functions are orthogonal to the Legendre block functions occupying the same octree cell, we can see this implies that the S_8 octree basis functions must have vanishing zeroth and first moments:

$$\int x_1^{\alpha_1} x_2^{\alpha_2} x_3^{\alpha_3} q_n(x_1, x_2, x_3) \mathrm{d}^3\boldsymbol{x} = 0, \tag{A4}$$

for $\alpha_i = 0, 1$ where $i = 1, 2, 3$ and the integral is over all space.

We can now define the Panphasia octree basis functions themselves using the functional forms defined above. Using the notation established in Section 2.2, we define a set of functions for each cell (j_1, j_2, j_3) in the octree at level l:

$$B^{l,n}_{j_1,j_2,j_3}(\boldsymbol{x}) = \frac{1}{\Delta_l^{3/2}} q_n\left(\frac{\boldsymbol{x} - \boldsymbol{x}^c(l, j_1, j_2, j_3)}{\Delta_l}\right), \tag{A5}$$

and $n = 1, 56$ labels the octree functional forms. The terms $\boldsymbol{x}^c(l, j_1, _j 2, j_3)$ and Δ_l give the cell centre and cell size defined in equations (11) and (10), respectively. The octree basis function obey the following normalization/orthogonality relations:

$$\int_{L^3} B^{l_1,n_1}_{j_1,j_2,j_3}(\boldsymbol{x}) B^{l_2,n_2}_{k_1,k_2,k_3}(\boldsymbol{x}) \mathrm{d}^3\boldsymbol{x} = \delta_{l_1 l_2}\delta_{n_1 n_2}\delta_{j_1 k_1}\delta_{j_2 k_2}\delta_{j_3 k_3}, \tag{A6}$$

where the integral is over the entire volume of the root cell. Because they are mutually orthogonal, as we have shown in Section 2, the expansion coefficients of a basis function expansion of a Gaussian white noise field are independent Gaussian variables. In Appendix B, we give the mapping between the pseudo-random number sequence and the basis functions that defines the Panphasia realization.

APPENDIX B: GENERATING A GAUSSIAN PSEUDO-RANDOM SEQUENCE AND MAPPING IT ON TO THE OCTREE

As described in Section 4 we use the MRGK5-93 generator to provide a very long periodic sequence of pseudo-random numbers r_i which are uniformly distributed between 0 and 1. We use the Box–Muller transformation (Box & Muller 1958), with a modification to generate a corresponding sequence of Gaussian pseudo-random numbers, g_i, with zero mean and unit variance:

$$\begin{aligned} g_{2i} &= \sqrt{-2\ln(r_{2i})}\cos(2\pi r_{2i+1}) \\ g_{2i+1} &= \sqrt{-2\ln(r_{2i})}\sin(2\pi r_{2i+1}). \end{aligned} \tag{B1}$$

A significant fraction of the computing time when evaluating Panphasia is spent in generating the Gaussian pseudo-random numbers. A faster method is described in Press et al. (1992), but as it uses the rejection method it is not suitable because we wish to establish a mapping for the entire available sequence without having to look at the pseudo-random number values themselves.

Table B1. Some reference values for Panphasia. The first column gives the position of a random number relative to the starting point. The sequence r_i are uniformly distributed pseudo-random numbers in the range $0 < r_i < 1$. The sequence g_i are corresponding Gaussian pseudo-random numbers, with zero mean and unit variance, and are generate in pairs using the Box–Muller transformation as described in the text of Appendix B.

i	r_i	g_i	State of pseudo-random number generator for r_i	Comment
0	0.716 483 784	0.536 408 766	(1538637210, 861452511, 1738028090, 1398591498, 1039141497)	Origin of sequence
1	0.864 066 010	−0.615 682 518	(1855567628, 1538637210, 861452511, 1738028090, 1398591498)	
4657 948	$6.915\,07 \times 10^{-8}$	4.574 061 225	(149, 1149276986, 1622633566, 1876117056, 1232329462)	Branch point
4657 949	0.101 988 118	3.411 353 097		
4657 949+2^{137}	$8.507\,921 \times 10^{-2}$		(182706232, 1864678143, 1322192784, 650896850, 1598221492)	$r_{\rm new}$

Equation (B1) does not work well for the rare occasions when r_{2i} is very small, as the pseudo-random numbers are discrete and there is a smallest non-zero pseudo-random number with a magnitude of about 2.3×10^{-10}. As the number of Gaussian variables in Panphasia is huge, the extreme tail of the Gaussian variables will be truncated if nothing further is done. To deal with this, we first check if $r_{2i} < 10^{-6}$, and if it is, then replace the value in equation (B1) with $r_{2i} = 10^{-6} r_{\rm new}$ where $r_{\rm new}$ is an alternate random number, whose origin will be discussed below. Should $r_{\rm new}$ be less than 10^{-6}, the procedure is repeated until a larger value is obtained.

The number $r_{\rm new}$ cannot be taken from the pseudo-random number sequence close to r_{2i}, as with the rejection method, it would interfere with the mapping between the octree and the pseudo-random number sequence, or the same number would be reused, which violates the requirement that the random numbers be independent. To get round this, $r_{\rm new}$ is computed by advancing the sequence from the position corresponding to r_{2i} by a very large and arbitrary shift of $2^{137} + 1$. This large shift guarantees in practice that the same random number is not used twice in making some particular initial conditions. This branching procedure is only really required for the even pseudo-random numbers, r_{2i}, as the odd values r_{2i+1} are used to calculate an angle which does not have singular behaviour at either end of the range. However, the code used for Panphasia applies the same branching conditions to both even and odd values although this makes a fairly negligible difference to the actual values of the Gaussian pair for the odd case.

This whole procedure is only enacted once in a million times. We tested that the modified routine does return a Gaussian distribution well into the tail of the distribution where the modification becomes important. The routine was also tested with a less stringent branch condition, $r_{2i} < 10^{-2}$, to ensure that it works if $r_{\rm new}$ is also small and one or more further iterations are required.

Having described the origin of the pseudo-random sequence, we move on to the mapping between this sequence and the octree. For each cell in the octree, there are 56 octree basis functions and we need to generate an expansion coefficient for each of these drawing the value from a Gaussian distribution with zero mean and unit variance. For reasons explained in Section 3, we also associate a further eight Gaussian random variables with each cell. These are not properly part of Panphasia but can be used if desired to generate an independent pseudo-random field. In total, 64 random numbers are needed for each octree cell.

For the ensemble average power spectrum on the scale of the root cell to be a white noise field, we need to add the effects of an infinite set of octree basis functions which are larger and overlap the root cell. This can be achieved simply by expanding the root cell in Legendre block functions. The first eight Gaussian pseudo-random numbers g_i, $i = 0, 7$, are reserved for the coefficients of these root-cell-sized Legendre block functions. After this every 64 consecutive Gaussian pseudo-random numbers are assigned to a particular octree cell, level by level, with increasing depth. For each cell, the first 56 pseudo-random numbers are assigned to the octree functions, while the final eight are not part of Panphasia and are available to generate a field with one value per octree cell that is independent of Panphasia. A raster scan pattern over the cells is used at every given level. Using the same notation for the octree cells as in the previous appendix, we define an integer function:

$$\phi(l, j_1, j_2, j_3) = 8 + 64 \left[4^l j_1 + 2^l j_2 + j_3 + \frac{8^{l-1} - 1}{7} \right], \tag{B2}$$

for $l > 0$. For octree cell (j_1, j_2, j_3) at level l of the octree, the first and last pseudo-random Gaussian variables for that cell are $g_{[\phi(l,j_1,j_2,j_3)]}$ and $g_{[\phi(l,j_1,j_2,j_3)+63]}$.

To define Panphasia we need also to specify the starting point of the pseudo-random number sequence. Table B1 gives the initial state of the random number generator plus some additional values which are useful cross-checks for anyone wanting to write their own code to generate Panphasia. The final three entries of the table show an example where the generation of the two Gaussian variables using equation (B1) requires the branching procedure described above to generate the final numbers.

Section 7 described how to publish the phases for a cuboidal patch within Panphasia. The final number in the Panphasian descriptor is a check number. This check number depends on the location of the cell in the octree, and also on an ASCII character string included in the descriptor. For a cuboid at level l of the octree with a corner nearest to the origin at (j_x, j_y, j_z) and side lengths d_x, d_y, d_z, we define three integers $I_1 = \phi(l, j_x + d_x - 1, j_y, j_z)$, $I_2 = \phi(l, j_x, j_y + d_y - 1, j_z)$ and $I_3 = \phi(l, j_x, j_y, j_z + d_z - 1)$.

The check number, $N_{\rm check}$, is given by

$$N_{\rm check} = \left(T^{I_1}(1) + T^{I_2}(1) + T^{I_3}(1) + \sum_{i=1}^{n} i T^{{\rm ascii(string}(i))}(1) \right) \bmod m, \tag{B3}$$

where T is the state vector of the random number generator, used in equation (32) and the sum is over the n characters given in the descriptor name. The function ascii() returns the ascii value of character. For example, ascii(A) = 65, ascii(a) = 97.

APPENDIX C: THE DEFINITION OF PANPHASIA

We now combine the results of the previous appendices to give a complete definition of the Panphasia field at a position $\boldsymbol{x} = (x_1, x_2, x_3)$, where $0 \leq x_i < L$, $i = 1, 2, 3$,

$$W_{\rm Panphasia}(\boldsymbol{x}) = \sum_{i_1=0}^{1}\sum_{i_2=0}^{1}\sum_{i_3=0}^{1} g_{4i_1+2i_2+i_3}\, p_{i_1 i_2 i_3}\left(\frac{2x_1-L}{L}, \frac{2x_2-L}{L}, \frac{2x_3-L}{L}\right) + \sum_{l=0}^{49}\sum_{j_1=0}^{2^l-1}\sum_{j_2=0}^{2^l-1}\sum_{j_3=0}^{2^l-1}\sum_{n=1}^{56} g_{[\phi(l,j_1,j_2,j_3)+n-1]} B^{l,n}_{j_1,j_2,j_3}(\boldsymbol{x}),$$

where all the symbols are defined as follows. The first term on the right-hand side is a sum over the eight Legendre block functions, $p_{j_1 j_2 j_3}$, defined in equation (13). The coefficients $g_{4i_1+2i_2+i_3}$ are Gaussian pseudo-random numbers with zero mean and unit variance, and are described in Appendix B. The summation in the second term on the right-hand side is over the entire set of octree basis functions. The index l denotes the level in the octree, where $l = 0$ corresponds to the root cell and increases with depth. The summations over j_1, j_2, j_3 are over the cubic cells making up level l of the octree. A description of the octree is given in Section 2. The sum over n is over the 56 orthogonal octree basis functions which occupy each cell and are defined in Appendix A. The sequence of Gaussian pseudo-random numbers, g, are linear within a cell and begin for a given cell at a location given by the integer function, $\phi(l, j_1, j_2, j_3)$, defined in equation (B2). The octree basis functions themselves, $B^{l,n}_{j_1,j_2,j_3}$, are defined by equation (A5).

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