The Astrophysical Journal Supplement Series, 252:28 (18pp), 2021 February https://doi.org/10.3847/1538-4365/abcd94

GenetIC—A New Initial Conditions Generator to Support Genetically Modified Zoom Simulations

Stephen Stopyra[1], Andrew Pontzen[1], Hiranya Peiris[1,2], Nina Roth[1,3], and Martin P. Rey[1,4]
[1] Department of Physics and Astronomy, University College London, London WC1E 6BT, UK; s.stopyra@ucl.ac.uk
[2] The Oskar Klein Centre for Cosmoparticle Physics, Department of Physics, Stockholm University, AlbaNova, Stockholm SE-106 91, Sweden
[3] Zurich Gruppe Deutschland, Deutzer Allee 1, D-50679 Köln, Germany
[4] Lund Observatory, Department of Astronomy and Theoretical Physics, Lund University, Box 43, SE-221 00, Lund, Sweden
Received 2020 June 11; revised 2020 October 6; accepted 2020 November 23; published 2021 February 4

Abstract

We present genetIC, a new code for generating initial conditions for cosmological *N*-body simulations. The code allows precise, user-specified alterations to be made to arbitrary regions of the simulation (while maintaining consistency with the statistical ensemble). These "genetic modifications" allow, for example, the history, mass, or environment of a target halo to be altered in order to study the effect on their evolution. The code natively supports initial conditions with nested zoom regions at progressively increasing resolution. Modifications in the high-resolution region must propagate self-consistently onto the lower-resolution grids; to enable this while maintaining a small memory footprint, we introduce a Fourier-space filtering approach to generating fields at variable resolution. Due to a close correspondence with modifications, constrained initial conditions can also be produced by genetIC (for example, with the aim of matching structures in the local universe). We test the accuracy of modifications performed within zoom initial conditions. The code achieves subpercent precision, which is easily sufficient for current applications in galaxy formation.

Unified Astronomy Thesaurus concepts: N-body simulations (1083)

1. Introduction

The generation of initial conditions is a crucial step in simulations of cosmological structure and galaxy formation. Simulation codes require as input the positions and velocities of dark matter and baryons at an early time in the universe's development, when deviations from homogeneity are approximately linear. The basic task is to generate a Gaussian random field, with a specific power spectrum, on a discrete grid that samples the continuous density-contrast field. This in turn can be used to generate velocity and displacement fields for a set of particles, and for mesh codes, fluid variables for each grid cell. Starting from Gaussian white noise, all these fields can be generated via a suitable series of convolutions.

A significant complication in performing these convolutions arises if we wish to work with zoomed initial conditions, where the grid spacing differs from one region of the simulation to another. Zoom simulations are attractive because they focus computational effort on a single object, allowing it to be modeled with far greater fidelity than is possible for a population; however, for initial condition generation, efficient convolution algorithms typically require a fixed grid resolution. One possibility is to generate the field at uniformly high resolution and downsample within the unzoomed majority of the domain (e.g., Katz et al. 1994; Navarro & White 1994; Tormen et al. 1997; Prunet et al. 2008). However, this becomes prohibitively wasteful of memory and processing resources as the desired dynamic range between zoom and volume increases. As a solution to this problem, GRAFIC2 (Bertschinger 2001) introduced an algorithm that generates fields directly on a nested grid structure, albeit with the need to pad out the high-resolution region to twice its final side length. A refined implementation of the same basic algorithm is provided by the widely used generator MUSIC (Hahn & Abel 2011).

These initial conditions generators have helped zooms become a standard technique for high-resolution galaxy formation studies. However, we often want to understand how a galaxy's observable properties have been affected by its history and local environment. This is impossible with a single object or even a small sample from a typical cosmological volume, given that each galaxy differs from its counterparts in multiple distinct respects. A "brute force" approach of generating many different random realizations of the field until we find one that looks sufficiently similar—yet with the desired differences—would be exceptionally time-consuming and imprecise. Consequently, an attractive alternative is to generate systematic variations in the accretion history, environment, or other aspect of a *single* object that one then resimulates several times. In this paper, we will describe a code (available at https://github.com/pynbody/genetIC) which generates and then minimally modifies initial conditions to make such "genetic modifications," while maximizing the likelihood for the field to have arisen as a random Gaussian draw from the ΛCDM power spectrum (Roth et al. 2016; Pontzen et al. 2017; Rey et al. 2019b).

Operationally, such modifications closely resemble the Hoffman–Ribak procedure for generating constrained initial conditions (Hoffman & Ribak 1991). Existing initial conditions codes that have the ability to perform Hoffman–Ribak constraints therefore can, in principle, be used for simple modifications as well. However, in existing implementations, the algorithm can only be applied to a single grid at once; when applied to a zoom simulation, modifications do not propagate correctly out of the high-resolution region, leading to discontinuities and errors in the correlation function. Moreover, an effective modification algorithm needs to target the field value averaged in regions of arbitrary shape, as determined by tracing the material in halos or their environment back to the initial conditions (Roth et al. 2016). To our knowledge, no public algorithm for performing such multiresolution or arbitrary shape manipulations currently exists.

In this paper, we present `genetIC`, a new code that implements solutions to these issues and so is suitable for generating zoomed, genetically modified initial conditions. As a fortuitous side effect of implementing multiresolution modifications, `genetIC` also drastically reduces the memory footprint for realizations on a given zoom geometry, due to near-elimination of the need for padding. At present, output can be made to `GADGET`, `TIPSY`, or `GRAFIC` formats (the latter being suitable for use with `RAMSES`). While not its primary focus, `genetIC` can also perform global field manipulations such as inversion and power spectrum fixing, which enable insights into the growth of large scale structure (e.g., Angulo & Pontzen 2016; Pontzen et al. 2016; Anderson et al. 2019). The code is modular and extensible so that additional manipulations or output formats can easily be added at a later date.

We review the generation of multiresolution initial conditions and explain the new implementation in `genetIC` in Section 2, with supporting mathematical derivations given in Appendix A. The core algorithm used for applying constraints consistently across resolution boundaries is described in Section 3, with the full details in Appendix A.4. We discuss the accuracy of `genetIC` with examples in Section 4. Section 5 then gives an overview of the structure of the code. A short summary is provided in Section 6.

2. Description of genetIC

2.1. Review of Generating Initial Conditions

At its heart, the problem that any initial conditions generator aims to solve is generating a set of N particles that approximate a smooth density field described by an initial density contrast, $\delta(\boldsymbol{x})$. Genetic modification, which we will describe in Section 3, seeks to manipulate this field in a manner consistent with a draw from the same Gaussian ensemble, as specified by the cosmological power spectrum.

The typical procedure is first to generate a discrete vector, $\boldsymbol{\delta}$, that samples $\delta(\boldsymbol{x})$ at a set of points $\boldsymbol{x}_i$ on a regular lattice. The vector $\boldsymbol{\delta}$ should be a random draw from a distribution with a covariance determined by the cosmology, i.e.,

$$\langle \boldsymbol{\delta}\boldsymbol{\delta}^\dagger \rangle = \mathsf{C}, \tag{1}$$

where C is a discrete version of the cosmological covariance matrix. To create $\boldsymbol{\delta}$ we start from a unit-variance, uncorrelated white noise field $\boldsymbol{n}$, and then multiply by $\mathsf{C}^{1/2}$.

Once generated, each element of the vector $\boldsymbol{\delta}$ gives the density contrast averaged over a particular grid cell of the simulation. The next task is to translate this into a corresponding set of particles with positions and velocities. This can be achieved using Lagrangian perturbation theory (see Buchert 1993; Buchert & Ehlers 1993). In this prescription, particles are labeled by their initial grid positions, $\boldsymbol{q}$, and their evolution is tracked using a displacement field, $\boldsymbol{\Psi}(\tau, \boldsymbol{q})$, that gives their position at some later time; the dynamics of $\boldsymbol{\Psi}(\tau, \boldsymbol{q})$ can be solved perturbatively. The lowest-order terms constitute the Zel'dovich (1970) approximation, which links velocities and displacements directly to gradients of the potential. Provided that we start at sufficiently high redshift (typically $z > 100$), the Zel'dovich approximation is adequate for galaxy formation questions for which `genetIC` is principally designed; however, the modular nature of `genetIC` allows the method used to be extended to higher order if required—see Section 5 for further discussion.

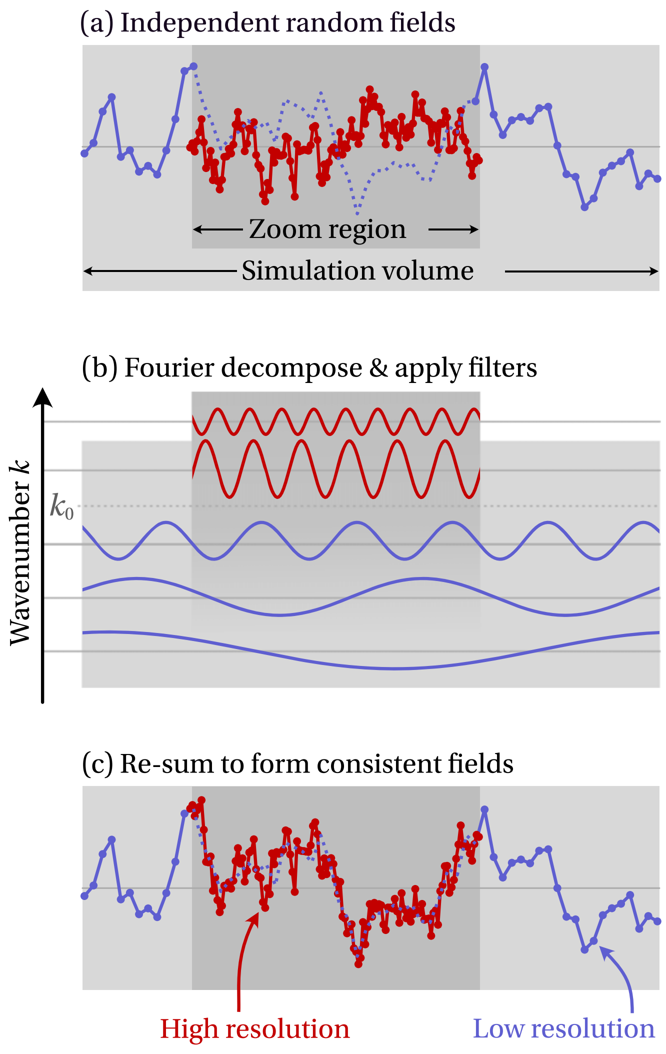
(a) Independent random fields
Zoom region
Simulation volume
(b) Fourier decompose & apply filters
Wavenumber k
k_0
(c) Re-sum to form consistent fields
High resolution
Low resolution

Figure 1. Illustration in 1D of zoom simulation initial conditions, as implemented by `genetIC`. A high-resolution grid is inserted as a "zoom region" into a low-resolution grid. (a) Both grids are initially seeded independently, but this generates long-wavelength modes in the high-resolution grid that are inconsistent with the low-resolution field. To solve this problem, we (b) split modes in Fourier space, as described in Section 2.3. Only high-frequency modes (wavenumber above k_0) are retained on the new grid. (c) We then replace the missing long-wavelength modes with an appropriately filtered version of those in the low-resolution grid.

2.2. Zoom Initial Conditions

The prescription described in Section 2.1 is complete for a uniform grid. However, most applications of genetic modifications will make use of zoom simulations incorporating the high resolution needed to simulate individual galaxies and halos, while accurately retaining the gravitational effects of a large-scale environment (e.g., Katz et al. 1994; Navarro & White 1994). This setup is illustrated in the bottom panel of Figure 1. The upper two panels show the field generation process in `genetIC`, which we will now motivate before describing in detail in Section 2.3. In the depicted example, there are two grids; a low-resolution grid covering the full simulation domain, and a high-resolution grid that covers the "zoom" region. Any approach to generating zoom initial conditions must solve the problem of how to relate random fields on the two grids in order to obtain the correct correlation structure between the two regions.

One possible way to generate a zoom simulation is to calculate the entire cosmological volume at high resolution, then degrade the sampling outside the target region. This is inefficient, however, and entirely infeasible for some applications; for example, in the EDGE suite (Rey et al. 2019a; Agertz et al. 2020), the initial conditions resolution corresponds to an effective grid of 32768^3. Storing a single such grid at double precision requires 32 terabytes; manipulating it is entirely out of reach. To obtain the high effective resolution of EDGE, a 1024^3 grid is instead nested inside two 512^3 grids. Each level has a physical extent four times smaller than the one above. A single field in this hierarchy requires only ten gigabytes to store, despite reaching the required effective resolution in the region of interest.

As stated above, the challenge is to ensure that the final, multiresolution field has the correct correlations despite being represented in a piecemeal manner. Specifically, we need the long-wavelength modes of the fine grid to match those of the low-resolution grid in the region where they overlap, and for the finite box size of the fine grid and the boundary between the two grids to have minimal impact.

The only solution to this problem currently described in the literature is presented by Bertschinger (2001) and Hahn & Abel (2011). It involves nesting a large buffer region around the section of the high-resolution grid before applying convolutions, in order to create the correct boundary conditions for the small grid, thus allowing long-wavelength convolutions to be carried out safely. These "ghost" regions expand the total storage requirements to 73 gigabytes per field (in the EDGE example given above). Note that the components of the output displacement field, $\boldsymbol{\Psi}(\tau, \boldsymbol{q})$, each count as a field, making the minimal memory requirements of such an algorithm almost 300 gigabytes for this scenario. While sufficiently powerful computer resources are available, once modifications are introduced (each one with their own associated field), the computational demands spiral further upward. Moreover, there is no existing algorithm describing how ghost regions should correctly interact with modifications to the field. This poses a problem since, to maintain continuity and consistency with the cosmological power spectrum, it is essential that the long-wavelength behavior of the modification propagates correctly outside the zoom region.

The two considerations above motivate finding a solution to multiresolution convolution and field modification that does not involve ghost regions. Our approach is to view the fundamental problem as one of band limiting: we wish to supplant the original low-resolution information with additional modes above the original Nyquist frequency.[5] From this perspective, the solution is to combine modes in Fourier space. Figure 1 outlines how this procedure starts from two independent random fields on the separate grids and combines them into a consistent multiresolution realization. The Fourier perspective also leads to an algorithm for applying field modifications across resolution boundaries that will be discussed in more detail in Section 3.

[5] The Nyquist frequency, $k_{\rm nyq}$, is defined to be half the sampling rate, i.e., $k_{\rm nyq} = N\pi/L$, where N is the number of grid points along one edge of the simulation box.

The MUSIC code (Hahn & Abel 2011) has an option to use Fourier-space filtering, blending modes between low- and high-resolution grids during convolution with the correlation function (Hahn, private communication). This has not been explicitly documented in the scientific literature, but was first implemented for the AGORA code comparison project (Kim et al. 2016) and is now switched on by default. The MUSIC scheme is not designed to reduce memory requirements, and so, compared to our approach, it has a different set of design considerations for its Fourier filters. We will describe below how genetIC filters are band-limited in Fourier space, but unlike in MUSIC, they also remain compact in real space.

2.3. The Fourier-space Approach

We will now describe in more detail how the procedure motivated above works in practice. We continue to focus on the case of two levels; nesting multiple subgrids requires recursively applying this two-level case.

The initial aim is to construct $\boldsymbol{\delta}_L$, a vector containing a low-resolution sampling of the density field on a coarsely pixelized grid, and $\boldsymbol{\delta}_H$, a high-resolution vector that stores information only in the zoom region. The challenge is to generate these starting from two independent white noise fields, $\boldsymbol{n}_L$ and $\boldsymbol{n}_H$, on the low-resolution and high-resolution grids, respectively.[6]

In our Fourier-space approach, we think of this problem as being closely related to constructing filtered versions of an underlying high-resolution field $\boldsymbol{\delta}$ for the whole grid to which we do not have access. Such filtered fields would take the form

$$\tilde{\boldsymbol{\delta}}_L = \mathsf{F}_L \boldsymbol{\delta}, \tag{2}$$

$$\tilde{\boldsymbol{\delta}}_H = \mathsf{F}_H \boldsymbol{\delta}, \tag{3}$$

where F_H and F_L are filters preserving high-frequency and low-frequency Fourier modes, respectively. By itself, this does not lead to a practical algorithm, since $\tilde{\boldsymbol{\delta}}_L$ and $\tilde{\boldsymbol{\delta}}_H$ consume the same space in memory as the original vector, $\boldsymbol{\delta}$. Instead, we want $\boldsymbol{\delta}_L$ to be a pixelized version of $\tilde{\boldsymbol{\delta}}_L$, defined on the low-resolution grid, and $\boldsymbol{\delta}_H$ to be a version of $\tilde{\boldsymbol{\delta}}_H$, which is confined to the high-resolution region only. By choosing F_L appropriately, $\tilde{\boldsymbol{\delta}}_L$ is band-limited even on the coarse pixelization and therefore can be losslessly "compressed" to low resolution. Storing $\boldsymbol{\delta}_H$ only in the zoom region can also be regarded as a compression, albeit one that explicitly destroys information which we do not require.

This Fourier-space decomposition suggests a practical three-step algorithm for generating $\boldsymbol{\delta}_L$ and $\boldsymbol{\delta}_H$:

1. Draw independent white noise fields for $\boldsymbol{n}_L$ and $\boldsymbol{n}_H$, and convolve with the theoretical cosmological correlation function independently on both grids to produce $\boldsymbol{\delta}_L$ and $\boldsymbol{\delta}_H$;
2. Apply the high-pass filter to $\boldsymbol{\delta}_H$ and the low-pass filter to $\boldsymbol{\delta}_L$;
3. Sum the two fields to produce the final field in the high-resolution region.

[6] The current version of genetIC uses independent white noise on each grid. An alternative would be to start from realizations that are already strongly correlated, so that the phase of high-frequency modes is not dependent on the placement of the zoom region. This can be arranged, for example, by producing white noise relative to an octree basis function (Jenkins 2013; Jenkins & Booth 2013).

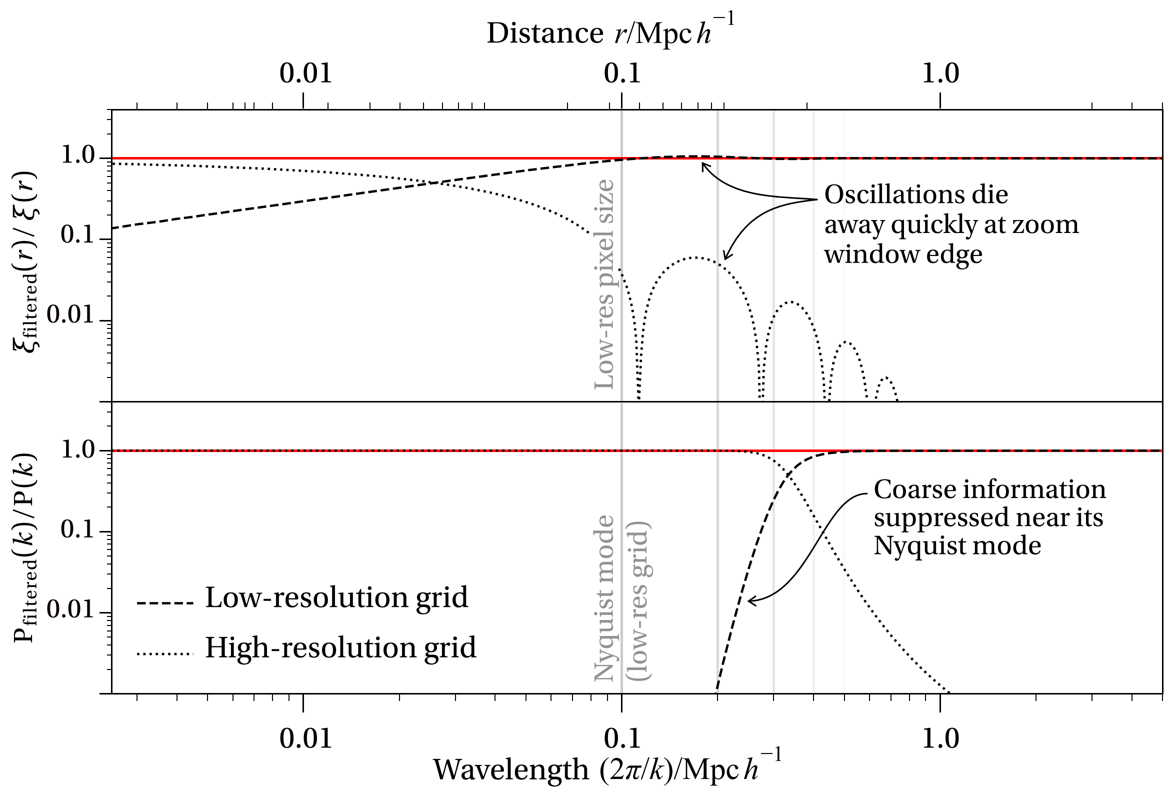

Figure 2. Top: ratio of the filtered to the unfiltered real-space correlation function, $\xi_{\rm filtered}(r)$, using the Fermi filter, Equation (4), in Fourier space. Axes are logarithmic, so we show the absolute value of the ratio. The contribution from the high-resolution correlations dies away quickly (within a few low-resolution pixels), meaning the low-resolution field carries most of the correlation structure. However, over the decaying region, there are some oscillations due to the trade-off required between the sharpness of the filter in Fourier space and real space (see Section 2.3 for further discussion). Bottom: ratio of the filtered to unfiltered power spectrum for the same scenario. Wavelength $2\pi/k$ is used so that scales always increase from left to right. The filter is chosen to ensure we exclude small-scale modes above the Nyquist frequency of the low-resolution grid, while simultaneously ensuring the fine-grid, real-space correlation function rapidly decays outside the high-resolution region. Vertical gray lines show the low-resolution pixel size (top panel) and equivalently, the Nyquist frequency (bottom panel) and its integer multiples.

This process is illustrated in Figure 1. We will discuss precisely how the fields from the two grids are combined in Appendix A. Note that, in step (a), convolving with the correlation function entails a trade-off between real-space and Fourier space accuracy (Pen 1997; Sirko 2005). For uniform-resolution volumes, this is a matter of preference (Orban 2013), but zoom regions generated with the Bertschinger (2001) algorithm are only self-consistent when using a real-space transfer function (Hahn & Abel 2011). The `genetIC` algorithm computes all long-wavelength correlations on the base grid, which permits self-consistent convolution with either a real-space or Fourier-space transfer function, as desired.

We now turn to the specific choice of filter. The filters F_L and F_H need to be designed such that $\boldsymbol{\delta}_L$ can be stored at the low pixel resolution without aliasing, while $\boldsymbol{\delta}_H$ can be stored in the high-resolution region and contains no large-scale correlations (so that the finite size of the high-resolution region does not impinge on any manipulations). These requirements are in tension but can be satisfied approximately with an appropriate Fourier space filter.

To balance these requirements `genetIC` uses a Fourier-space Fermi–Dirac distribution, as illustrated in Figure 2:

$$\mathsf{F}_L(k) = (\exp[(k - k_0)/(k_0 T)] + 1)^{-1}. \tag{4}$$

The high-resolution filter, F_H, is then related to this by

$$\mathsf{F}_L^2 + \mathsf{F}_H^2 = \mathbb{I}, \tag{5}$$

which is required in order to recover $\boldsymbol{\delta}$ from the sum of the filtered fields $\mathsf{F}_L \tilde{\boldsymbol{\delta}}_L + \mathsf{F}_H \tilde{\boldsymbol{\delta}}_H$, or equivalently, to obtain the correct power spectrum in the high-resolution region (see Appendix A.2). The wavelength at which the cutoff in the filter must occur should be larger than the Nyquist length, but smaller than the zoom region length scale. There will inevitably be some trade-off in the choice of the "temperature," T: the filter must remain sufficiently smooth in Fourier space that oscillations in the real-space filter die away rapidly, but an excessively smooth filter will attempt to retain information at or above the Nyquist frequency, leading to inaccurate small-scale correlations; see Figure 2.

The default code choices are $T = 0.1$ and $k_0 = 0.5\, k_{\rm nyq}$, where $k_{\rm nyq}$ is the Nyquist frequency of the low-resolution grid. These were selected based on experimentation with the practical performance of the algorithm in a 1D test setting where exact covariances can be computed. The cosmological power spectrum is red on small scales ($k > 0.2h\,{\rm Mpc}^{-1}$) but blue on large scales ($k < 0.2h\,{\rm Mpc}^{-1}$); therefore, we tested a range of power laws and the actual dimensionless cosmological spectrum (i.e., maintaining the total power per unit log wavenumber when moving from the 3D to the 1D setting). With our choice above, we found that fractional errors on covariances of the final field are always smaller than ~2% (Figure 3). In Figure 3, we compare with the "traditional" approach, which involves using a large padding region around

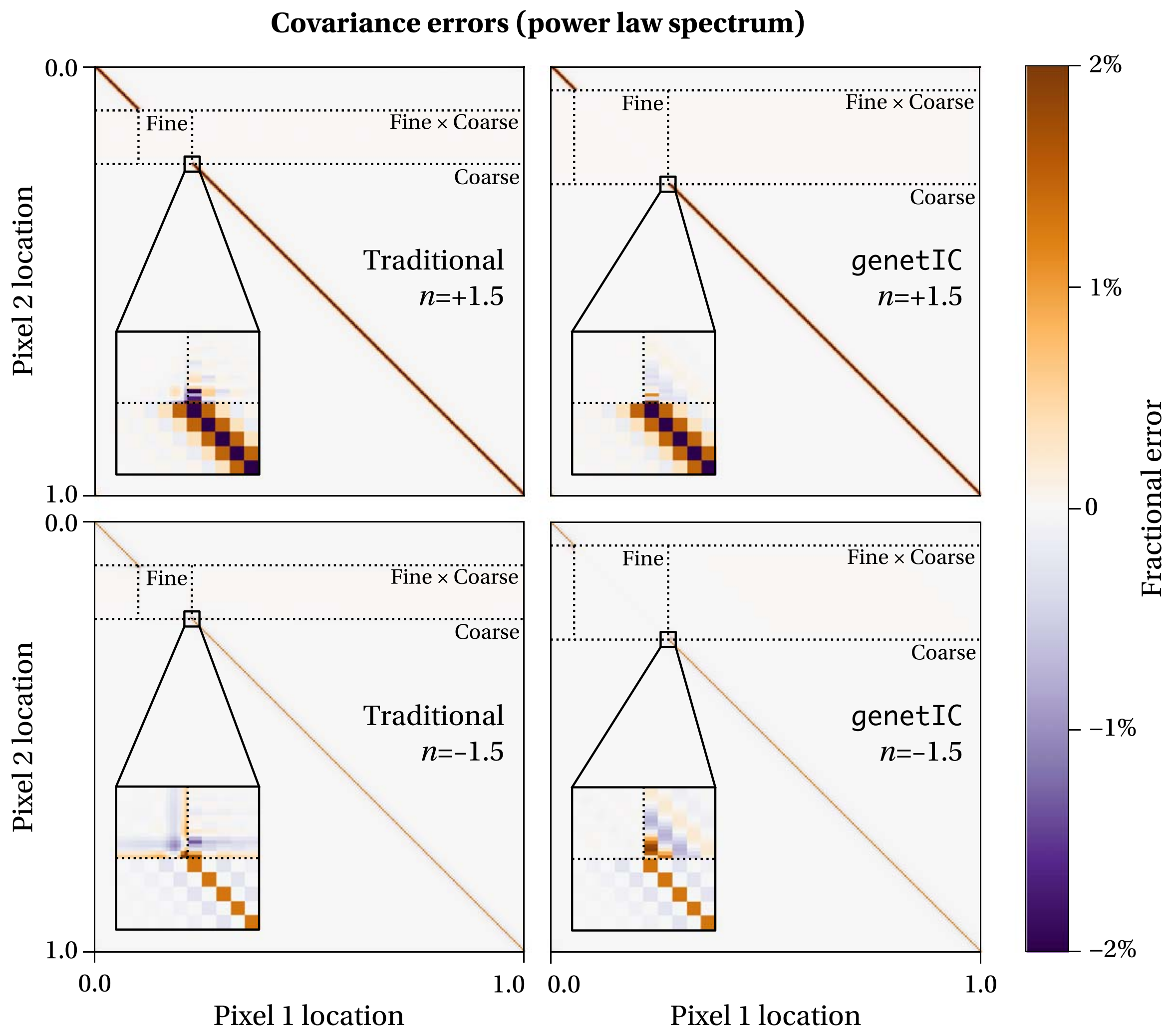

Figure 3. Errors on the covariance matrix of zoom initial conditions for a 1D toy example of genetic modification. The base grid here is defined on a line between 0 and 1, and each panel shows the error in covariance between different pixels as a matrix. Two example 1D power spectra, $P(k) \propto k^{1.5}$ (top panels) and $P(k) \propto k^{-1.5}$ (lower panels), are tested. Errors for traditional (left panels) and Fourier filtering (right panels) approaches are shown relative to the idealized covariance matrix obtained by making realizations at high resolution across the whole box and then degrading the low-resolution region. We have fixed the computational demands, meaning that only half the high-resolution box is available for use in a traditional approach (the remainder being required as padding). Inset panels show the covariance error in the region around the transition from low-resolution to high-resolution pixelization. In both cases, the largest errors are found throughout the low-resolution box, due to power above the band limit. This error is present and unavoidable in all cosmological simulations. All other errors are small (of order 1%), making either approach acceptable for practical application to galaxy formation simulations.

the outside of the high-resolution box (resulting in a smaller usable high-resolution volume for the same computational effort compared to `genetIC`).

The dominant error term is aliasing on the low-resolution grid, which reflects the existence of power above the Nyquist limit. This error is irreducible and present in *all* cosmological simulations, although the reddening of the spectrum at high k means it rapidly becomes smaller at higher resolutions. Other errors are comparable to (or smaller than) this dominant term, and are similar in magnitude between the Fourier space and traditional approaches. For this reason, we have not explored alternative filtering methods, although we note that (should increased accuracy ever be required) a multiresolution wavelet (e.g., Daubechies 1992) approach to combining information from different zoom levels may be worth investigating.

3. Modifications for Zoom Simulations

The most important feature of `genetIC` is the generation of modified initial conditions. Fundamentally, we wish to map an existing field, $\boldsymbol{\delta}$, onto a new field, $\boldsymbol{\delta}'$, that satisfies a chosen condition but is otherwise minimally altered. If the density field is stored at a single resolution for the whole simulation and the condition is described by a linear function of the density field, then the solution is provided by the Hoffman–Ribak algorithm (Hoffman & Ribak 1991).

In Hoffman–Ribak applications, the original, unconstrained field typically is treated as an intermediate object and not used for any computation. In modifications, on the other hand, the unconstrained field is treated as physically meaningful; we must simulate the unmodified galaxy as well as a series of modified versions in order to probe how galaxies react to their surroundings or cosmological history. For modifications described by linear maps of the original density field, the resulting procedure is otherwise equivalent to the Hoffman–Ribak procedure. However, nonlinear modifications result in a different procedure (Rey & Pontzen 2018).

In the case of a zoom simulation, it is not immediately clear how one should make modifications that are consistent across the boundaries between different resolution grids. We will next discuss broadly how this problem is solved in `genetIC`, and give full details of the algorithm and its derivation in Appendix A.4.

3.1. Linear Modifications

To provide the starting point for modifications in zoom simulations, let us first describe more fully the approach for a uniform-resolution grid. In the simplest possible scenario, we want to change the average value of the field over some region, either multiplying it by a constant or setting it to a given value. Supposing the region corresponds to N elements of the density field, δ_{i_1}, δ_{i_2} … , δ_{i_N}, the constraint can be expressed as

$$\frac{1}{N}(\delta_{i_1} + \delta_{i_2} + \dots + \delta_{i_N}) = \bar{\delta}, \tag{6}$$

where $\bar{\delta}$ is the target average density contrast we wish to impose. Because such modifications are defined in terms of a linear sum of elements of the density-contrast vector, $\boldsymbol{\delta}$, we refer to them as *linear modifications*. Because the potential and velocity fields are themselves linearly related to the density, any linear modification can be expressed as a constraint on the density field. In the general case, the constraint would be described by a vector $\boldsymbol{u}$ and the target value d, and we aim to achieve

$$\boldsymbol{u} \cdot \boldsymbol{\delta}' = d. \tag{7}$$

In the average-density example, $\boldsymbol{u}$ is a vector with components $1/N$ for elements of $\boldsymbol{\delta}'$ that lie in the region to be modified, and zero outside it, and $d = \bar{\delta}$.

The Hoffman–Ribak algorithm uses $\boldsymbol{u}$ and d to create a map from the unmodified field, $\boldsymbol{\delta}$, to the modified field, $\boldsymbol{\delta}'$:

$$\boldsymbol{\delta}' = \boldsymbol{\delta} + \frac{(d - \boldsymbol{u} \cdot \boldsymbol{\delta})\mathsf{C}\boldsymbol{u}}{\boldsymbol{u} \cdot \mathsf{C}\boldsymbol{u}}. \tag{8}$$

One can verify that $\boldsymbol{\delta}'$ satisfies Equation (7) by applying the dot product with $\boldsymbol{u}$ to both sides. However, this is not the only way of satisfying the constraint—Equation (8) is a special choice because it can be regarded as finding the solution $\boldsymbol{\delta}'$ that minimizes $(\boldsymbol{\delta}' - \boldsymbol{\delta})^\dagger \mathsf{C}^{-1}(\boldsymbol{\delta}' - \boldsymbol{\delta})$, subject to the constraint of Equation (7). Consequently, modifications made this way are *minimal* and are the most likely way of satisfying the constraint that could have arisen from a Gaussian random field with correlation matrix C. This disfavors, for example, unphysical modifications such as sharp discontinuities in the density field. See Roth et al. (2016) and Rey & Pontzen (2018) for further discussion.

The method can be used to force extreme modifications to the field; for example, two neighboring individual pixels might be set by the user to take wildly different values, violating the expected continuity. In this case, the solution to Equation (8) will be equally extreme: for example, a power spectrum estimate based on the single resulting field may differ significantly from the ΛCDM ensemble mean, and non-Gaussianity null tests may fail. Such rare deviations are always present in any Gaussian ensemble, but have a large χ^2 reflecting their rarity. `GenetIC` helps users understand whether they are generating outliers by outputting the χ^2 change between the modified and unmodified fields. Provided the change is of order unity or less, the new realization can be considered a similarly likely draw from the Gaussian random ensemble as the original unmodified field.

The procedure leading to Equation (8) is suitable for a simulation with a uniform resolution over the whole simulation grid. We now explain how the procedure generalizes to the case of multiple resolutions combined using the filters from Section 2.3. We may assume that the modification is specified on the highest-resolution grid, since by construction this grid will contain the galaxy that we wish to alter. However, it is incorrect to apply the modification only on this high-resolution grid.

To understand why, consider a modification that changes the average value in some region Γ that lies entirely within the highest-resolution grid. First note that Γ, the set of particles that define a modification, is not the same as the set of particles that will be modified by it. We might imagine changing only the field within this region in order to match the constraint, but doing so would either be impossible in the overdensity due to mass conservation, or for any other quantity, would produce a discontinuity at the edge of Γ. Accordingly, it cannot be a *minimal* modification: a field with sharp discontinuities is highly unlikely to arise from a random draw from a Gaussian distribution.

As discussed above, one of the outcomes achieved by the Hoffman–Ribak algorithm is that the modification avoids such unphysical realizations by choosing a form that maximizes the likelihood for the modification to have been drawn from a Gaussian distribution. However, the result is that the field outside of the initially specified region Γ is also modified, and in particular, the necessary changes in the field $\boldsymbol{\delta}$ will propagate out to regions outside the highest-resolution grid even if the set of particles that defined the modification lie entirely within it. If we only modify the highest-resolution grid, then there will be a discontinuity at the grid's edge—which again would be unphysical, or at the very least, highly unlikely within the ΛCDM ensemble. Making the high-resolution grid sufficiently large that the modification is negligible outside of it is infeasible for the computational performance reasons discussed in Section 2.2. The modification must instead simultaneously alter the high- and low-resolution grids in a way that is self-consistent.

Our approach is to return to the idealized defining relations of $\tilde{\boldsymbol{\delta}}_L$ and $\tilde{\boldsymbol{\delta}}_H$, Equations (2) and (3), and their compressed versions $\boldsymbol{\delta}_L$ and $\boldsymbol{\delta}_H$. The defining relations can be used to deduce the required operations on $\boldsymbol{\delta}_L$ and $\boldsymbol{\delta}_H$ that are equivalent to applying the Hoffman–Ribak procedure to the underlying high-resolution field $\boldsymbol{\delta}$. To achieve this, we concatenate $\boldsymbol{\delta}_L$ and $\boldsymbol{\delta}_H$ into a single compressed vector $\boldsymbol{\delta}_Z$ (Z here standing for "zoom").

Modifications form a map $\boldsymbol{\delta} \to \boldsymbol{\delta}'$ taking the underlying uncompressed density field into its modified counterpart. An ideal solution for the compressed modified field, $\boldsymbol{\delta}'_Z$, would be obtained by first modifying the field $\boldsymbol{\delta} \to \boldsymbol{\delta}'$, then compressing $\boldsymbol{\delta}' \to \boldsymbol{\delta}'_Z$. For a practical algorithm, we are searching for a single operation that modifies $\boldsymbol{\delta}_Z \to \boldsymbol{\delta}'_Z$ in an equivalent way. In other words, we want the operations of modification and compression to a zoom simulation to commute, at least approximately.

Given this principle, we can derive the required operation on the zoom vector $\boldsymbol{\delta}_Z$ by regarding the operation of compression, mapping $\boldsymbol{\delta} \to \boldsymbol{\delta}_Z$ as a coordinate transformation.[7] Then, the Hoffman–Ribak algorithm, whose application is straightforward for a fixed-resolution vector $\boldsymbol{\delta}$, can be transformed into an operation defined in this "zoom basis" by the standard rules for performing coordinate transformations on linear operators. This results in a well-defined map from $\boldsymbol{\delta}_Z$ to a new vector $\boldsymbol{\delta}'_Z$, which satisfies the required constraint as if it had been performed at high resolution everywhere and then degraded to the zoom simulation (to the level of precision that a given zoom setup can describe). Having established this approach of demanding consistency with the original algorithm, the modifications can be applied at almost any stage of the initial conditions generation process (see Section 2.3). In the final code, we choose to apply the modifications as early as possible, prior even to convolution with the transfer function. The full details are derived and explained in Appendix A.4.

3.2. Other Types of Linear Modification

The prescription for modifying density contrasts is easily extended to modifying other quantities linearly related to the density contrast, such as velocity and the gravitational potential. For example, the first-order gravitational potential, Φ, satisfies the Poisson equation, which in Fourier space, assuming matter domination, is

$$\Phi_k(\boldsymbol{k}) = -\frac{3\Omega_m H_0^2}{2a}\frac{\delta_k(\boldsymbol{k})}{k^2}, \tag{9}$$

where the subscript k denotes Fourier space quantities, $\Omega_m = \rho_m/\rho_{\rm crit}$ is the current matter density relative to the critical density, H_0 is the Hubble constant, and a is the scale factor when the initial conditions are generated. Because the potential perturbation is linearly related to the density contrast, we can represent a constraint on the potential as a different constraint on the density contrast in the case of uniform resolution. One starts by constructing the vector describing the region in which the potential is to be modified, just as in the density case; see Equation (6). Next, the Poisson Equation (9) is applied to $\boldsymbol{u}$, allowing the Hoffman–Ribak algorithm to modify the potential while still operating on the overdensity field (van de Weygaert & Bertschinger 1996). In the case of zoom simulations, one cannot perform this operation directly, since solving the Poisson equation implies a convolution that leaks information from the high-resolution into the low-resolution region; instead, we again use the uniform-resolution case as an idealized limit and thus derive the correct procedure for zooms as described above in Section 3.1.

Velocity modifications are implemented starting from the linear relationship between velocity and density contrast in the Zel'dovich approximation. Specifically, velocity perturbations are proportional to the gradient of the potential; during matter domination, the Fourier space relationship is given by

$$v_j(\boldsymbol{k}) = -i\Omega_m^{1/2}H_0 a\frac{k_j\delta_k(\boldsymbol{k})}{k^2}. \tag{10}$$

[7] Strictly speaking this transformation is not invertible—we address how to deal with this in Appendix A.4.

Once more, the uniform-resolution algorithm is obtained by applying the velocity transformation to the $\boldsymbol{u}$ vector; the variable-resolution case is again derived by requiring agreement with the idealized limit.

3.3. Quadratic Modifications

All of the modifications we have discussed up to this point are linear. Another type of modification consists of quadratic modifications to the field. These were first discussed by Rey & Pontzen (2018) and amount to modifying the density-contrast vectors to satisfy

$$\boldsymbol{\delta}^{\dagger\prime}\mathsf{Q}\boldsymbol{\delta}' = q, \tag{11}$$

where Q is a matrix. The simplest example of such a modification is altering the variance of an arbitrary region within the simulated density field. This can be useful, for example, in altering the smoothness of an overall halo merger history. The solution for uniform-resolution fields is already described in Rey & Pontzen (2018), and consists of a gradient descent approximation involving a local linearization of the problem. Because each step in the gradient descent is approximated by a Hoffman–Ribak update, no extra work is required to implement the algorithm in zoom simulations. The `genetIC` code allows variance modifications to be requested by the user, and internally transforms these into an appropriate sequence of linear modifications. Constructing an appropriate Q operator allows other types of quadratic modifications to also be computed in the future, and `genetIC` is modularized to allow for these to be added easily.

4. Accuracy and Examples

To illustrate the accuracy of `genetIC` in performing convolutions across resolution boundaries, we will now verify the accuracy of its output when modifying zoom initial conditions. This can be accomplished by comparing to an ideal equivalent set of initial conditions that are realized at uniformly high resolution (and then degraded outside the zoom region). Studying the accuracy of these modifications complements our earlier investigation of the real-space correlation function generated when using the code's Fourier-split approach (Figure 3).

4.1. Accuracy of the Initial Conditions

The setup is as follows: we generate a 200 Mpc h^{-1} cubic box at $z = 99$ and resolution $N_{\rm sim} = 512^3$, as the "high-resolution everywhere" initial conditions. These play the role of the "ideal" case, which for higher-resolution zoom simulations would be infeasible to run—at this relatively low resolution, however, we can compare the "ideal" and zoom simulations directly. The zoom simulation is defined on the same size of box, but with a 128^3 low-resolution grid and a zoom window consisting of a 50 Mpc h^{-1} cube of resolution $N_{\rm window} = 128^3$ centered on $\boldsymbol{x} = (50, 50, 50)$ Mpc h^{-1}. Note that its spatial resolution therefore matches that of the original 512^3 box. Both sets of initial conditions are seeded with the same random seed in Fourier space. Because the low-resolution modes are seeded first, they match between the simulations; however, we caution that, because the Fourier modes have different meanings between the idealized and the zoom case, there is no way to

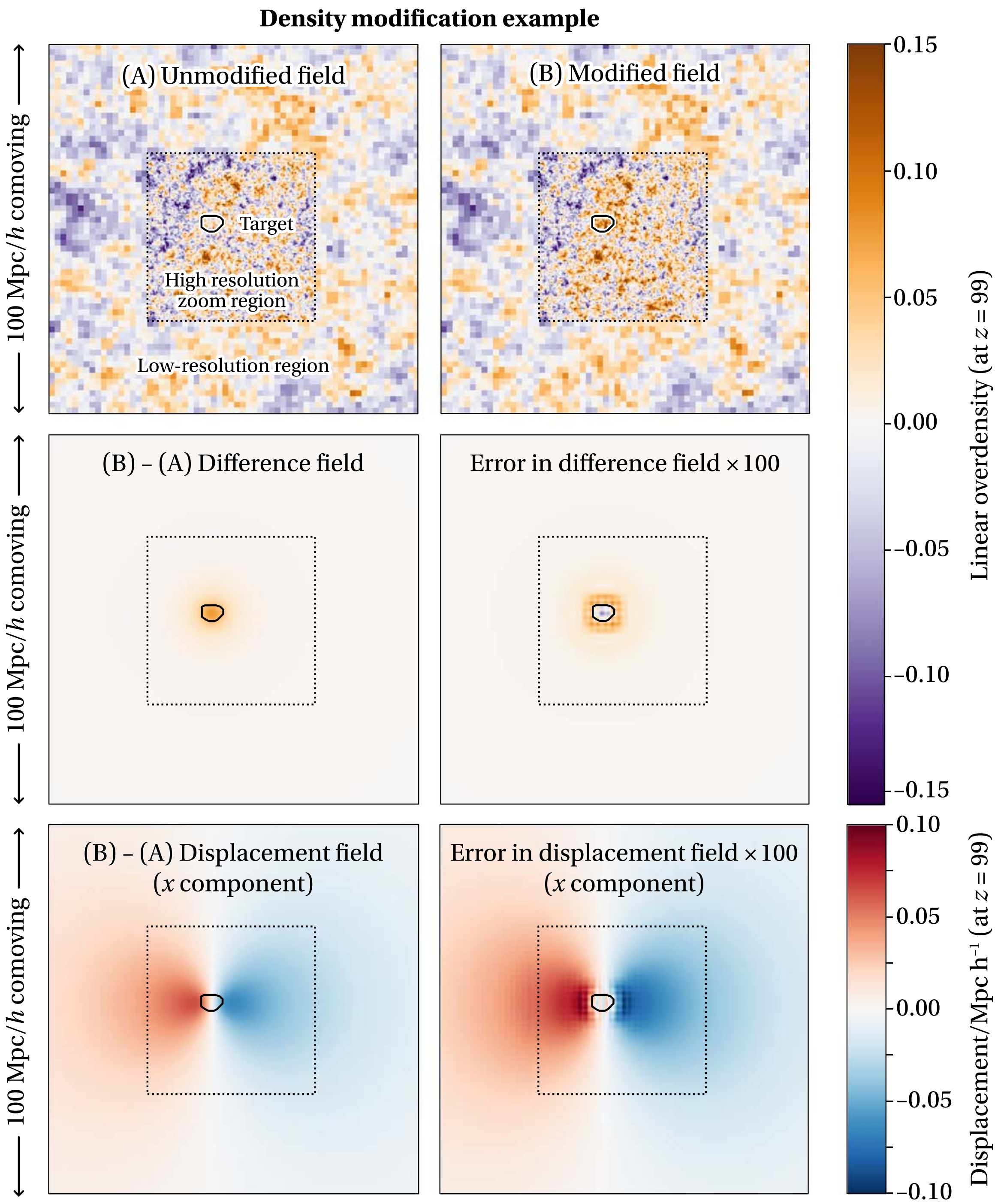

Figure 4. Density slices illustrating the modification of a halo's Lagrangian volume in a zoom simulation (128^3, 50 Mpc h^{-1} high-resolution grid embedded within a subvolume of a 128^3 low-resolution grid that is 200 Mpc h^{-1} on a side; here, only the central 100 Mpc h^{-1} region is shown). We change the average density contrast in the region indicated by the loop, while the dotted square represents the boundary of the high-resolution region. Middle left plot shows the difference between modified and unmodified fields, which is smooth across the resolution boundary (for a clearer example of this continuity, see Figure 5). In the middle right panel, we compare the difference field to that which is obtained in the idealized case where the whole volume is realized at high resolution, plotting the difference defined in Equation (13) exaggerated by a factor of 100. We obtain percent-level accuracy relative to the idealized case. Bottom panels are the same as the middle panels, but for the x component of the displacement field instead of the density field.

make the high-resolution modes match exactly in this test. This will necessitate a scaling in the comparison, which we will describe in due course.

An example slice through initial conditions on this grid is shown in the top left panel of Figure 4. The set of particles to be modified, Γ, is the Lagrangian region of a particular halo, chosen by evolving forward the unmodified zoom initial conditions, selecting a halo in the zoom window, and tracking its particles back to the initial conditions (see Roth et al. (2016) and Section 5.4 for a description of this process, which is unique to `genetIC`). A density-contrast modification then makes minimal alterations to the field in order to change the density contrast averaged over those particles. For the purposes of this example, we will choose the constraint on the modified field, $\boldsymbol{\delta}'$, to be

$$\langle \boldsymbol{\delta}' \rangle_\Gamma = 0.1. \tag{12}$$

This describes a particularly large modification, comparable to the rms of the entire field, for the purposes of illustration. The effect on the zoom initial conditions is shown in the top right panel of Figure 4. In the middle left panel, we show the difference between the modified and unmodified fields (with the same color scale as the overdensity field itself). To characterize the error in this modification relative to the "ideal" case, we subtract the difference between the zoom simulation modification and a rescaled ideal modification

$$\Delta(\boldsymbol{x}) = [\delta'_{\rm Zoom}(\boldsymbol{x}) - \delta_{\rm Zoom}(\boldsymbol{x})] - \alpha[\delta'_{\rm Ideal}(\boldsymbol{x}) - \delta_{\rm Ideal}(\boldsymbol{x})], \tag{13}$$

where $\delta_{\rm Ideal}(\boldsymbol{x})$ and $\delta_{\rm Zoom}(\boldsymbol{x})$ represent the field as computed in the idealized and zoom cases, respectively, and primes indicate modified fields. The factor α rescales the ideal modification field. This is needed to make a proper comparison because the premodification average of the density-contrast field in any given region is not the same for idealized and zoom fields, that is, $\langle \boldsymbol{\delta}_{\rm Zoom} \rangle_\Gamma \neq \langle \boldsymbol{\delta}_{\rm Ideal} \rangle_\Gamma$: the fields unavoidably have different high-resolution modes simply due to the way that random noise is seeded. Consequently, the modification fields $\tilde{\boldsymbol{\delta}}_{\rm Ideal} - \boldsymbol{\delta}_{\rm Ideal}$ and $\tilde{\boldsymbol{\delta}}_{\rm Zoom} - \boldsymbol{\delta}_{\rm Zoom}$ that enforce Equation (12) have different amplitudes. The rescaling factor needed to compare them is just the ratio of these modification amplitudes:

$$\alpha = \frac{\langle \tilde{\boldsymbol{\delta}}_{\rm Zoom} \rangle_\Gamma - \langle \boldsymbol{\delta}_{\rm Zoom} \rangle_\Gamma}{\langle \tilde{\boldsymbol{\delta}}_{\rm Ideal} \rangle_\Gamma - \langle \boldsymbol{\delta}_{\rm Ideal} \rangle_\Gamma}. \tag{14}$$

The final result of computing the error, Equation (13), is plotted in the middle right panel of Figure 4, exaggerated in scale by a factor of 100 relative to the fields and modifications themselves, showing that the error in modifications is vastly smaller than the modifications themselves. Since modifications in practical scenarios will generally be much smaller than the example presented here, this small error on already small modifications will be negligible. Crucially, the modification field (defined as the difference between the modified and unmodified fields) is smooth across both the boundary of the set of particles defining the modification and the grid boundaries (middle left panel of Figure 4). We also show, in the bottom panels of Figure 4, the difference between the modified and unmodified displacement field (x component, bottom left), and the error in the same displacement field magnified by a factor of 100 (bottom right). This illustrates the smoothness of the actual velocities and particle displacements that will affect the resulting particle distribution directly.

The continuity across the boundary is better seen by making a longer-wavelength, high-amplitude modification. In Figure 5, we consider a modification to the velocity field, obtained by constraining the average over Γ of the x component of the velocity to be $100\,{\rm km\,s^{-1}}$. Because of the additional k^{-1} weighting of Equation (10), the corresponding modification affects wavelengths up to the fundamental mode of the box. (Velocity correlations in cosmological simulations extend up $\simeq$1 Gpc if the box size is sufficiently large.) The modification is accurately propagated outside the zoom window, with only small errors compared to performing the same modification on a higher-resolution grid. It should be noted that the errors in the top and bottom right panels are again magnified by a factor of one hundred—although these highlight discontinuity in the errors, the overall modification field is still accurate and continuous to better than percent level.

4.2. Accuracy of Evolved Examples

The discussion in Section 4.1 demonstrates the accuracy of `genetIC`'s approach to convolutions that transfer information from small to large scales. However, it is also important to establish that large-scale information is correctly propagated into zoom regions; of particular concern is the persistence of structure when creating zoom simulations and modifying them. To demonstrate this persistence, we ran a $100\,{\rm Mpc}\,h^{-1}$, 512^3, dark-matter-only "base" simulation with $\Omega_m = 0.3156$, $h = 0.6726$, $\sigma_8 = 0.830$ (Planck Collaboration et al. 2016). We then generated a second set of "zoomed" initial conditions that introduced eight times higher mass resolution within a $20\,{\rm Mpc}\,h^{-1}$ cube at the center. Because both simulations are generated with the same random seed, the low-resolution modes of both match. Both base and zoomed simulations were evolved with `GADGET2` (Springel 2005).

We show in Figure 6 the evolved state at $z = 0$ in the selected region; the left panel shows a projection of a 5 Mpc slice through the original simulation, and the center panel shows the zoomed simulation, with the dashed line showing the approximate boundary of the zoom region in projection. The large-scale structure and locations of resolved satellite halos are preserved, while additional small-scale structure is introduced as expected. The mass of individual resolved structures changes by less than 1%.

We then selected a halo in the zoom region and increased the density contrast of its constituent particles in the initial conditions by 20%. This increased the mass of the final halo from $0.88 \times 10^{13}\,M_\odot\,h^{-1}$ to $1.78 \times 10^{13}\,M_\odot\,h^{-1}$, as expected (Roth et al. 2016). The evolved, modified density field is shown in the rightmost panel of Figure 6. The modifications to the halo result in small changes to its surrounding structures, maintaining consistency with the ΛCDM power spectrum. These changes include small shifts in the location of structures, some of which are perpendicular to the projected slice (which means structures can disappear from the figure). However, the overall large-scale structure is maintained, as expected.

5. Code Overview and Core Features

We now give a technical overview of the `genetIC` code. There are three main stages involved in a typical scenario when using `genetIC`: (1) grid creation and white noise generation; (2) modification; and (3) particle generation. These stages are illustrated in Figures 7 and 8, together with the code classes involved listed underneath each step of the stages. We do not describe the user syntax for controlling these stages through a parameter file, since these are detailed in a separate manual. The code is implemented in C++, with parallelization via OpenMP. It relies on the GNU Scientific Library (Gough 2009) for random number generation and FFTW for Fourier transforms (e.g., Frigo 1999).

5.1. Stage 1—Grid Setup and White Noise Generation

In the first stage (see Figure 7), the parameters are used to construct a `Grid` object that stores properties of the simulation, such as the cell size, particle mass, and dimensions. To enable zoom simulations, refinement grids are set up according to the

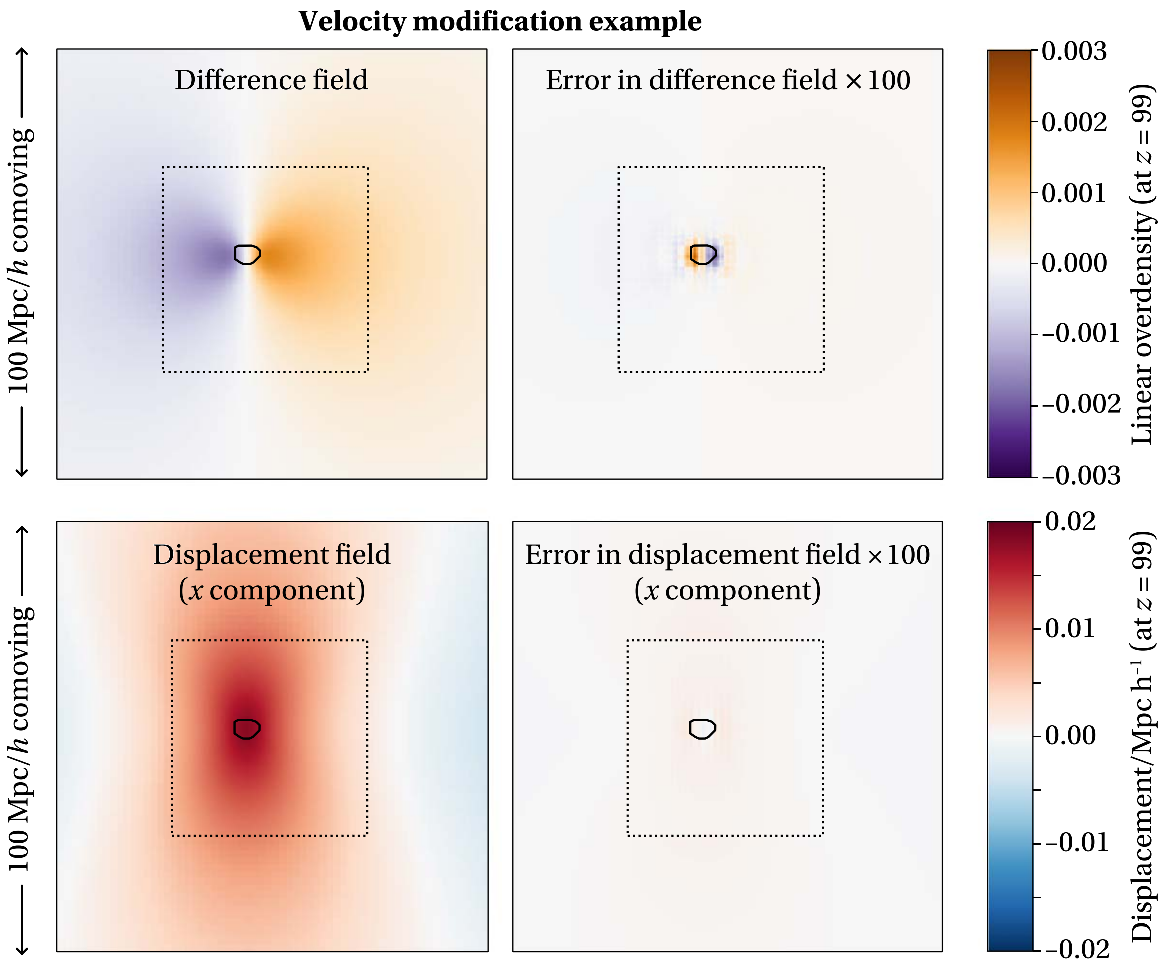

Figure 5. Top left: difference between modified and unmodified density-contrast field after a velocity modification on the same grid as Figure 4 (128^3 low-resolution 200 Mpc h^{-1} grid, 128^3 high-resolution 50 Mpc h^{-1} region, indicated by the dotted square). As with density-contrast modifications, the difference between modified and unmodified fields (left) is smooth across the boundary of the grids; this is now more clearly demonstrated, because velocities are correlated on larger scales. Top right: error in this difference compared with the same modification on a higher, fixed-resolution grid (512^3 across the full 200 Mpc h^{-1} box), exaggerated by a factor of 100. This exaggeration highlights a small discontinuity at the boundary between grids with different resolutions, but we emphasize that the error remains below percent-level. Bottom left: changes to the displacement field (which, in the Zel'dovich approximation, is directly proportional to the velocity). Bottom right: error in these changes compared with the same modification on the higher, fixed-resolution grid, exaggerated by a factor of 100. The magnitude of the errors in this case are too small to be visible.

specification in the parameter file and organized into a `MultiLevelGrid` class that encapsulates the relationship between different grids. It stores data about each level's size, grid layout, and position, as well as various functions for accessing and manipulating these grids. Any part of the code needing access to the relationship between different grids uses this object.

At the same time, a `ParticleMapper` object is set up to track how the grids are related to particles: for example, if the user specifies that a sphere around some point of the simulation should be stored at higher resolution, then a grid is set up that contains this sphere and the `ParticleMapper` keeps track of which particles in that sphere were requested and should be included at the particle generation stage (see Section 5.3). The `ParticleMapper` is able to map bidirectionally between grid cells to particles, an essential facility in the modification stage below. Performing this mapping can be expensive in complicated geometries, but is parallelized for maximum efficiency.

Once the grids are initialized, the code constructs a `MultiLevelField` object that contains `Field` objects, each of which stores the overdensity field on a single `Grid`. A `RandomFieldGenerator` uses the random seed specified in the parameter file to generate unit variance white noise within each `Field`. The `RandomFieldGenerator` makes use of GNU Scientific Library's Ziggurat algorithm (Gough 2009) to draw uncorrelated Gaussian samples, starting from a user-supplied seed integer (this allows the same set of random numbers to be drawn, irrespective of system architecture). Separating this module into its own object allows the random number generator back end to be easily changed, and there are several options for how the random components of the fields can be generated. In particular, the random draws can be performed both in Fourier and real space, and either in series or in parallel, according to the user's requirements. When drawing in parallel, the work is segmented into Fourier-space shells that can be drawn independently, so the final result is independent of the number of threads available.

Note that it is also possible to import pre-existing white noise fields, such as those that might have been generated by other initial conditions generators, instead of drawing random noise. These must be in `numpy` format, and must be of the correct size to fit the grids to be generated. This allows the possibility of applying genetic modifications to existing simulation suites.

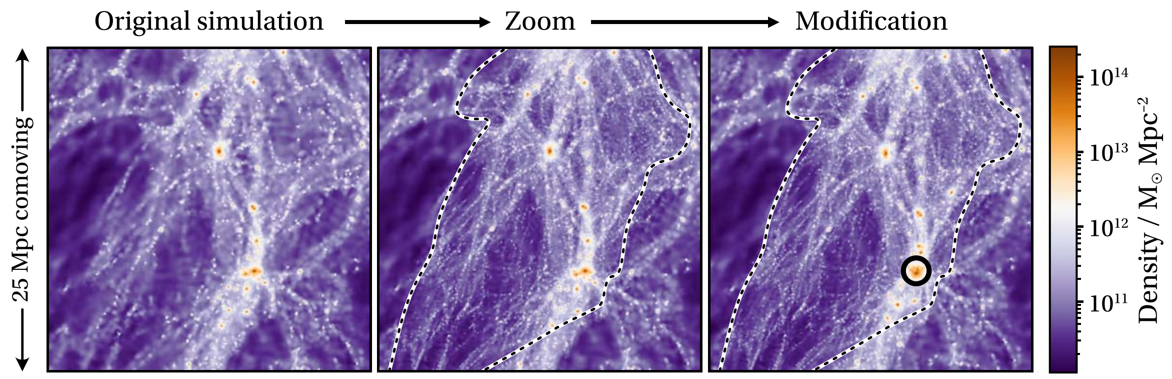

Figure 6. Example of `genetIC` in use. Left: density in a 5 Mpc-thick slice through a 100 Mpc h^{-1} uniform-resolution simulation, evolved to $z = 0$ using `GADGET2`. Center: same density slice taken through a simulation in which we have created a large cubic "zoom" region with eight times better mass resolution. (Zoom region deforms away from a cube due to nonlinear evolution, and its boundaries are shown as a dashed line.) Large-scale structures persist between the unzoomed and zoomed simulation, and increased detail can be seen in the zoomed version. Right: halo (circled) within the zoom region is selected and genetically modified to increase its initial density by 20%, and the simulation is performed once again. The result is to increase the halo's $z = 0$ mass from $8.73 \times 10^{12}\, M_\odot\, h^{-1}$ to $1.78 \times 10^{13}\, M_\odot\, h^{-1}$ (over a factor of two increase) while making minimal changes to the surrounding structure. Note that, due to small movements perpendicular to the page, small halos can move in and out of our slice—and thus spuriously appear or disappear in the right panel.

5.2. Stage 2—Modifications

In the second stage (see Figure 7), modifications are applied. These are specified in the parameter file and can be a linear or quadratic function of a set of particles (such as the average density). The `ParticleMapper` is used to trace the target particles into appropriate cells within the `MultiLevelGrid` structure. The specified modifications are used to construct `LinearModification` and `QuadraticModification` objects as appropriate, which are stored by the `ModificationManager`—this object is the heart of the `genetIC` algorithm. It implements all modification algorithms, and applies them to the white noise field.

The linear modifications currently implemented are density, potential, and velocity in three Cartesian directions. They are each represented by `LinearModification` subclasses, thus allowing future expansion or custom modifications to be defined. For quadratic modifications, only variance is currently implemented, and it is in a subclass of `QuadraticModification` so that future expansion should be straightforward.

5.3. Stage 3—Particle Generation

In the final stage, the modified white noise fields are converted into particle positions and velocities. This step is illustrated in Figure 8. First, the power spectrum is applied to the white noise field on each grid; however, this does not result in the final overdensity, since the filtering and mode combination step (Section 2.3 and Appendix A.2) has not yet been undertaken. The `MultiLevelParticleGenerator` object takes responsibility for the required steps, as well as producing the displacement and velocity fields that are required for the simulation initialization. The latter are generated by applying the appropriate convolution, such as Equation (10), on the individual grids independently, and then filtering and combining the modes in the fields just as for the overdensity field. While the code currently assumes the Zel'dovich approximation during these manipulations, other methods, such as higher-order Lagrangian perturbation theory, could be implemented in the future by adding subclasses.

Finally, the code must map the fields as stored on the grids onto output particles. There are two interrelated aspects of this: (1) the correspondence between particles and grid cells of the field, and (2) the selection of a resolution for each region. To output particles while taking account of these considerations requires bringing together information in the `MultiLevelParticleGenerator` and the `ParticleMapper`. For example, consider selecting a sphere of a given radius around a specific point (or a more abstract region, such as the particles surrounding a particular halo). If we want to output these particles at high resolution, the code internally creates a cubic grid around them and generates a white noise field for the whole cube. However, for efficiency in the final simulation, we only want to output those particles explicitly selected for high resolution. We might also wish to insert a thin layer of "intermediary" particles that interpolate between the high and low resolution, or embed further zoom grids hierarchically. The `ParticleMapper` keeps track of this information and determines for each point whether we should output a low-resolution particle or a group of high-resolution particles, allowing `genetIC` to handle detailed nested geometries seamlessly while shielding other parts of the code from these complexities.

Additionally, a `ParticleEvaluator` object takes responsibility for outputting the appropriate fields for each grid. Depending on the precise simulation being used, as well as the options specified, different derived classes of the `ParticleEvaluator` are used. In particular, the code can generate particles at lower resolution than the underlying random fields (subsampling), or at higher resolution via interpolation (supersampling). These facilities are used when generating intermediary regions (see above) and are also available for explicit invocation by advanced users who wish to fine-tune the performance of a simulation. Further details can be found in the code and its documentation.

In summary, the `MultiLevelParticleGenerator` takes the field and turns it into the raw information for particles, by default using the Zel'dovich approximation. The `ParticleMapper` keeps track of particles to be generated as output, while the `ParticleEvaluator` keeps track of how the particles are related to the underlying density fields. To generate output, the `ParticleMapper` iterates over the particles to be generated, accessing the relevant density fields via an intermediary `ParticleEvaluator`. The mapper is also used to identify and trace cells corresponding to a list of particles from a previous output, as we now describe.

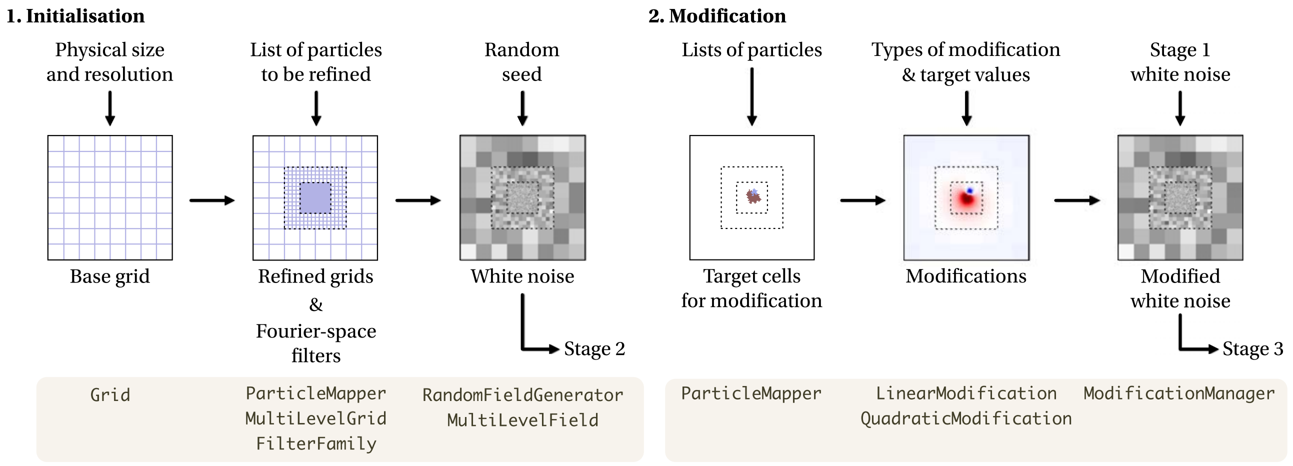

Figure 7. To provide an overview of the initial conditions generation process within `genetIC`, we break it down into three stages; here, the first two are illustrated. Control passes from left to right. Names of the classes involved at each step are given below each stage, and inputs flow from above. In step one, information specified by the user in a parameter file is used to construct a set of grid objects. If zoom levels are required, refinement grids are constructed. White noise fields are then generated for each grid. The second step only occurs if modifications are required, using information in the parameter file to construct and apply these to the white noise. Step three is illustrated in Figure 8.

5.4. Tracing Particles

A key feature of `genetIC` is its ability to map particles from a simulation back to an initial conditions grid cell. In some cases, this is a straightforward operation: if we perform a single unzoomed simulation, the mapping from the final particle index to the original grid position is simple. This can be used, for example, to select which grid cells will be refined when constructing a zoom simulation. However, if we then wish to generate a third set of initial conditions, in which we modify a region of the zoom simulation, the mapping is now considerably more complex, since only a fraction of the high-resolution grid is represented as particles. Additional complexities arise if baryon particles are added in a subregion; if a second zoom level is opened; if intermediate-resolution "padding" particles are added around a high-resolution region; or if genetic modifications change the set of particles that ultimately fall into the target galaxy.

One simple way around the mapping problem is to require that the simulation code associates an ID with each particle, with the ID uniquely identifying the origin grid cell, regardless of which cells are represented in the particle output. However, some codes do not offer this option; moreover, this kind of noncontiguous particle labeling can pose significant challenges for simulation post-processing and analysis routines.

The `genetIC` code can instead map from sequential particle IDs back to the original grid squares, when used with Lagrangian codes such as `GADGET` and `ChaNGa`. The user is required to specify the original parameter file that details how the original simulation was set up. By creating a second `ParticleMapper` object to process the input particles IDs, the code is then able to trace the relationship between particles and grid cells.

5.5. Baryon Transfer Functions

In *N*-body simulations, it is often assumed that, because dark matter is the dominant matter component, the difference between the baryon and dark matter transfer functions can be neglected and the total matter transfer function used for both. At late times, this is a good approximation (see Peebles 1980; Lyth & Stewart 1990). However, at early times, the baryon transfer function contains features not present in the dark matter power spectrum—most importantly, the effect of baryon acoustic oscillations. As cosmological parameters become determined more and more precisely, there may be a need in the future to take this difference into account.

For this reason, `genetIC` includes an option to take into account both the baryon and dark matter transfer functions in generating the initial conditions. This is accomplished by taking a copy of the white noise field after modifications but before applying the transfer function, and thus requires additional memory. However, the use of separate transfer functions is optional, and the user must explicitly enable it; otherwise, by default, the initial conditions use only the dark matter transfer function even for gas particles.

5.6. Paired and Fixed Initial Conditions

It is also possible to generate paired and fixed initial conditions. Paired fields have opposite phase in Fourier space, and so some correlations cancel between them, reducing sample variance while retaining Gaussianity. The effect is to swap overdensities and underdensities, which in itself may be useful for contrasting the growth of halos and voids (Pontzen et al. 2016). Fixed fields, on the other hand, set the power spectrum to the exact theoretical mean—and thus destroy Gaussianity in a controlled way (Angulo & Pontzen 2016); their properties are discussed further in Villaescusa-Navarro et al. (2018).

6. Discussion

We have presented `genetIC`, an initial conditions generator for cosmological simulations. The code generates multilevel zoomed initial conditions and is specifically designed to perform modifications to these initial conditions in a controlled manner.

`GenetIC` uses a new Fourier-space filtering approach to combine information from different resolution regions of the initial conditions (Section 2). This avoids the need for large ghost regions around each level of the simulation, enabling deeper and higher-resolution zooms; it also allows the code to self-consistently propagate information about modifications

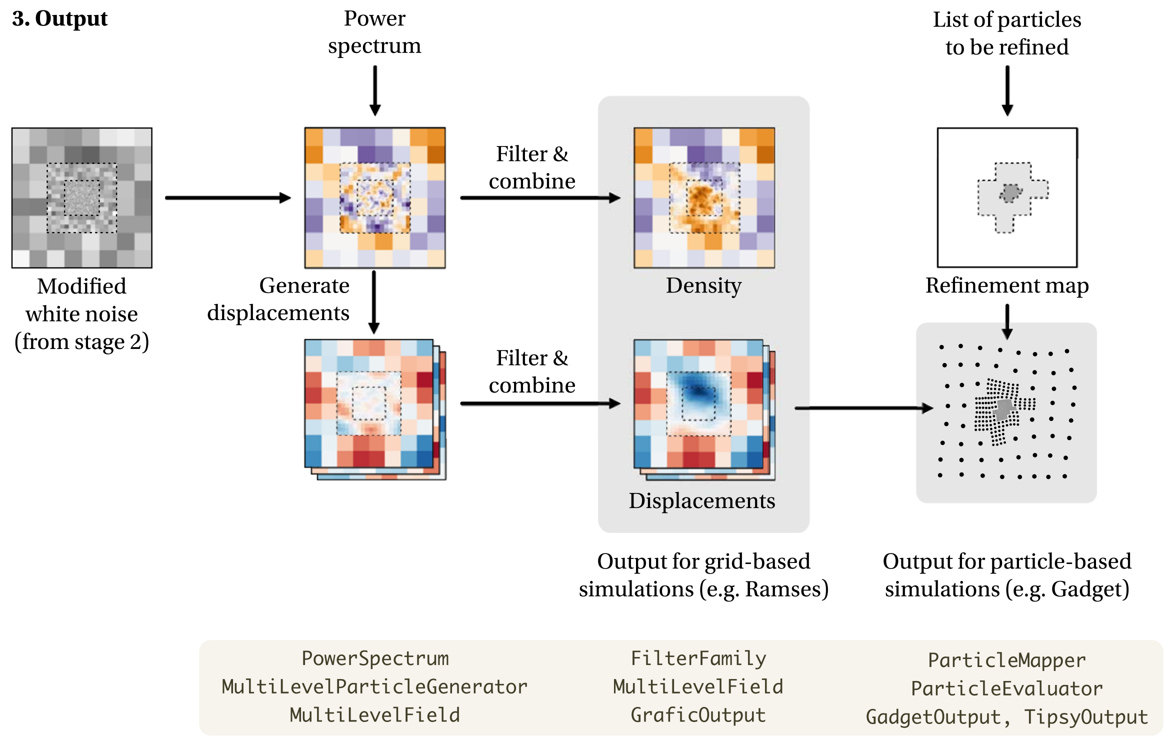
3. Output
Power spectrum
List of particles to be refined
Filter & combine
Modified white noise (from stage 2)
Generate displacements
Density
Refinement map
Filter & combine
Displacements
Output for grid-based simulations (e.g. Ramses)
Output for particle-based simulations (e.g. Gadget)
PowerSpectrum
MultiLevelParticleGenerator
MultiLevelField
FilterFamily
MultiLevelField
GraficOutput
ParticleMapper
ParticleEvaluator
GadgetOutput, TipsyOutput

Figure 8. Stage three of `genetIC`. As in Figure 7, classes for each step are indicated below, and inputs above. The (modified) white noise on each level has the power spectrum applied in Fourier space and is used to construct displacement fields for the particles. At this stage, the short-wavelength behavior of zoom grids is incorrect, i.e., the zoom regions do not match the long-wavelength modes of the base grid. This is corrected by filtering and combining the displacement fields on each level to give the correct long- and short-wavelength behavior in all zoom regions. Finally, the displacement fields are converted into particles or gridded output, ready for use with a simulation code.

made on the highest-resolution grid up to the lower-resolution grids.

The algorithms also by construction implement minimal modifications (Section 3), i.e., modifications that satisfy given constraints while minimizing changes and maximizing the likelihood of a given realization to have arisen from a Gaussian distribution. This prevents modifications from possessing unphysical features, such as sharp discontinuities at the boundary of regions that are modified. We verified that our implementation of multiresolution initial conditions produces correlation functions and modifications that are in close agreement with idealized uniform high-resolution equivalents (Section 4).

The code implements these concepts in an efficient parallelized manner (Section 5). In order to enable the specification of complex geometries for zoom regions and modifications, it has a sophisticated mapping system that ingests lists of particle IDs from prior simulations and identifies the associated grid cells for further manipulation (Section 5.1). The code has a variety of useful additional features, such as generating gas with the correct baryon transfer function (Section 5.5) and generating reversed initial conditions for the same initial seed (Section 5.6). One of its strongest features, however, is its modular, object-oriented design, which allows for easy extensibility to apply the code to different situations. A test suite accompanies the code to verify installations and ensure code quality. Support for the code and download links can be found at https://github.com/pynbody/genetIC. An extensive user manual is available from the github site, and the code is documented with Doxygen.

We gratefully acknowledge helpful comments and suggestions from the referee, Oliver Hahn. We would also like to thank Corentin Cadiou, Luisa Lucie-Smith, and Chris Pedersen for useful feedback. This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 818085 GMGalaxies. S. S. and A. P. were supported by the Royal Society. S. S., N. R. and H. V. P. were partially supported by the European Research Council (ERC) under the European Community's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement number 306478-CosmicDawn. H. V. P. was additionally supported by the research project grant "Fundamental Physics from Cosmological Surveys" funded by the Swedish Research Council (VR) under Dnr 2017-04212. M. R. acknowledges support from the Perren Fund, the IMPACT fund, and the Knut and Alice Wallenberg Foundation. This work was performed in part at the Aspen Center for Physics, which is supported by National Science Foundation grant PHY-1607611. This work was also partially supported by a grant from the Simons Foundation, and by funding from the UCL Cosmoparticle Initiative.

Appendix A
Mathematical Description of the GenetIC Algorithm

In Sections 2.3 and 3, we qualitatively described the approach implemented by `genetIC` to obtain and manipulate initial conditions for zoom simulations. We now provide a full analytic description, including further justification and derivations where appropriate.

We start by laying out some notation in Appendix A.1. Appendix A.2 then describes the relationship between white noise that is "compressed" (i.e., realized at variable resolution) and a uniform-resolution density-contrast field. Approximations required to build a practical algorithm are outlined in Appendix A.3. Finally, Appendix A.4 describes how we apply modifications. For completeness, Appendix B provides further technical insight into approximations.

As in the main paper, we assume throughout these appendices that there is a single zoom region, allowing us to make the notation as clear as possible. The code does, in fact, support nested regions where the resolution increases on each successive level, but this general multilevel case follows from recursion on the two-level approach.

A.1. Notation: Pixelization and Windowing

To express our construction of zoom initial conditions as accurately as possible, it is useful to introduce some notation. We will consider two key operations: pixelization and windowing. Conceptually, these both start from a field sampled at uniformly high resolution across the simulation domain, and respectively downsample to low resolution or extract the zoom region.

We start in each case from the vector $\boldsymbol{\delta}$, which contains a list of N field values for each pixel in the simulation when sampled at uniformly high resolution. In practice, we never generate fields in this way, since the goal is to avoid storing or manipulating such a prohibitively large vector (Section 2.1). However, the operators are nonetheless needed for describing and justifying the algorithm in the remainder of the Appendix.

The first operation, pixelization, will be denoted by P. It downsamples $\boldsymbol{\delta}$ to a low-resolution vector $\boldsymbol{\delta}_P$ of length N_P, i.e.,

$$\boldsymbol{\delta}_P = \mathsf{P}\boldsymbol{\delta}. \tag{A1}$$

Explicitly, P can be represented by an $N_P \times N$ rectangular matrix. The simplest resampling scheme is to create each low-resolution pixel by averaging over $m = N/N_P$ high-resolution pixels. For example, in the case of $m = 3$, we can visualize the P matrix as

$$\mathsf{P} = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots \\ & \vdots & & & \vdots & & & \ddots \end{pmatrix}. \tag{A2}$$

The other key operation is windowing, which selects a subset of the $\boldsymbol{\delta}$ vector corresponding to the region in which we wish to retain high-resolution initial conditions:

$$\boldsymbol{\delta}_W = \mathsf{W}\boldsymbol{\delta}. \tag{A3}$$

The matrix W has dimensions $N_W \times N$. The simplest explicit example in this case is given when the pixels in the zoom region are already located at the beginning of the original vector $\boldsymbol{\delta}$ (which, in fact, we may assume without loss of generality). Taking $N_W = 3$ for illustrative purposes, one would have

$$\mathsf{W} = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 1 & 0 & \cdots \end{pmatrix}. \tag{A4}$$

Pixelization and windowing both destroy information, and so cannot be inverted, but we will make use of their pseudo-inverses P^+ and W^+. These respectively upsample low-resolution field vectors to high resolution (in a specific way to be described shortly), and place the zoom region back into a full-sized simulation using zero-padding. They are defined to satisfy

$$\mathsf{P}\mathsf{P}^+ = \mathbb{I}_{N_P} \quad \text{and} \quad \mathsf{W}\mathsf{W}^+ = \mathbb{I}_{N_W}, \tag{A5}$$

where $\mathbb{I}_n$ is the $n \times n$ identity matrix. In other words, upsampling then downsampling or zero-padding then removing the zero padding must have no effect.

From this requirement, one may derive explicit expressions for P^+ and W^+. In the latter case, we have simply $\mathsf{W}^+ = \mathsf{W}^\dagger$, since $\mathsf{W}\mathsf{W}^\dagger = \mathbb{I}$; this is most easily verified by inspection of the example (A4), which generalizes to any N_W.

Deriving P^+ takes a little more care; recall that the pixelization P forms each low-resolution pixel from averaging over m high-resolution pixels. The pseudoinverse simply places that mean value back into each high-resolution pixel, leading to the expression $\mathsf{P}^+ = m\mathsf{P}^\dagger$. One may verify by inspection that, for the example given in (A2), $m\mathsf{P}\mathsf{P}^\dagger = \mathbb{I}_{N_P}$ as required; this generalizes to any value of m.

Finally, we note that the operators $\mathsf{P}^+\mathsf{P}$ and $\mathsf{W}^+\mathsf{W}$, describe respectively downsampling then upsampling the field and extracting the zoom region then zero-padding. These are both destructive operations but satisfy the projection relations

$$(\mathsf{P}^+\mathsf{P})^2 = (\mathsf{P}^+\mathsf{P}) \quad \text{and} \quad (\mathsf{W}^+\mathsf{W})^2 = (\mathsf{W}^+\mathsf{W}). \tag{A6}$$

Thus, there is no additional effect from repeated upsampling and downsampling or windowing and zero-padding. All the above relations are used routinely in derivations.

One can also construct operators that pixelize only in the window region, denoted P_W, or window the pixelized field, denoted W_P (as well as their pseudo-inverses, P_W^+ and W_P^+). The order between pixelizing and windowing does not matter, so

$$\mathsf{P}_W\mathsf{W} = \mathsf{W}_P\mathsf{P}. \tag{A7}$$

Unlike P and W, which are introduced as derivation tools rather than for practical computation, P_W and W_P can actually be implemented in practice. Operations that are performed by `genetIC` include upsampling the low-resolution information into the high-resolution (which can be notated as $\mathsf{P}_W^+\mathsf{W}_P$) and downsampling the high-resolution region into a zero-padded low-resolution full volume ($\mathsf{W}_P^+\mathsf{P}_W$).

According to the description above, P_W^+ can be thought of as a zero-order or "nearest neighbor" interpolation from low to high resolution. For some purposes, we will wish to use higher-order interpolation to form smoother fields from the underlying coarse pixelization. For brevity, we do not make a heavy distinction between different orders of interpolation in this appendix, but we do note that `genetIC` in practice uses a tricubic interpolation scheme, similar to that outlined by Lekien & Marsden (2005). For consistency, `genetIC` also implements downsampling P_W using an interpolation scheme that maintains $\mathsf{P}_W{\mathsf{P}_W}^+ = \mathbb{I}$.

A.2. Relationship between Variable-resolution and Uniform-resolution Fields

Existing algorithms for performing modifications, including the Hoffman & Ribak (1991) algorithm resulting from the formulation in Roth et al. (2016), are framed in terms of a uniform pixelization. We therefore need to derive a robust but computationally tractable algorithm for modifying zoom initial conditions. To start, we require an explicit analytic form for the map from white noise at varying resolution to a density-contrast field sampled at high resolution across the full simulation domain. This map can be regarded as a definition from which all algorithms derive, and is given for `genetIC` by

$$\boldsymbol{\delta} = \mathsf{C}^{1/2}\mathsf{F}_H\mathsf{W}^+\boldsymbol{n}_H + m^{-1/2}\mathsf{C}^{1/2}\mathsf{F}_L\mathsf{P}^+\boldsymbol{n}_L. \tag{A8}$$

In words, to obtain a density-contrast field at high resolution throughout the box, we would (i) place or resample the separate white noise fields into high resolution across the full volume (W^+ and $m^{-1/2}\mathsf{P}^+$, respectively); (ii) apply the appropriate high-pass F_H or low-pass F_L filters[8]; and (iii) convolve with $\mathsf{C}^{1/2}$ to finally obtain an appropriate covariance.[9]

For compact notation, one can concatenate the two noise vectors $\boldsymbol{n}_L$ and $\boldsymbol{n}_H$ into a single vector $\boldsymbol{n}_Z$ (where Z stands for "zoom"), and write

$$\boldsymbol{\delta} = \mathsf{T}\boldsymbol{n}_Z, \quad \text{where} \mathsf{T} = (\mathsf{C}^{1/2}\mathsf{F}_H\mathsf{W}^+ \quad m^{-1/2}\mathsf{C}^{1/2}\mathsf{F}_L\mathsf{P}^+). \tag{A9}$$

Thus, the transformation matrix T is an $N \times (N_P + N_W)$ matrix. We never explicitly calculate $\boldsymbol{\delta}$, or indeed, T, since these are prohibitively large in realistic simulations; its form is required only in the derivation of algorithms.

Given any matrix transformation M that can be applied to a uniform resolution volume $\boldsymbol{\delta}$, we would ideally wish to find an equivalent matrix M_Z that applies to the variable-resolution white noise, such that

$$\mathsf{M}\mathsf{T}\boldsymbol{n}_Z = \mathsf{T}\mathsf{M}_Z\boldsymbol{n}_Z, \tag{A10}$$

for any vector $\boldsymbol{n}_Z$. In general, there is not an exact solution to this problem, because T is noninvertible, meaning that M_Z is overdetermined. However, a uniquely well-motivated approximation is obtained by taking

$$\mathsf{M}_Z = \mathsf{T}^+\mathsf{M}\mathsf{T}\ , \tag{A11}$$

where T^+ is the pseudoinverse of T. The motivation for this expression can be understood at three levels of detail:

1. It is the most obvious generalization of the standard matrix transformation $\mathsf{M}_Z = \mathsf{T}^{-1}\mathsf{M}\mathsf{T}$ to the case where T is noninvertible.
2. In the case that $\mathsf{T}^\dagger\mathsf{T}$ is invertible, expression (A11) can be derived exactly from (A10), using the pseudoinverse identity $\mathsf{T}^+ \equiv (\mathsf{T}^\dagger\mathsf{T})^{-1}\mathsf{T}^\dagger$.
3. In the case of interest, where neither T nor $\mathsf{T}^\dagger\mathsf{T}$ are expected to be invertible, the expression can instead be derived using a maximum likelihood principle. This is shown in Appendix B.1.

In order to compute M_Z from M and hence formulate a practical implementation of modifications in zoom simulations, we will need an explicit expression for T^+. It is easiest to understand its form by bearing in mind that, practically speaking, T^+ maps from the uniform resolution $\boldsymbol{\delta}$ back onto varying-resolution white noise fields $\boldsymbol{n}_Z$. With this in mind, one can guess at an approximate solution,

$$\boldsymbol{n}_Z = \mathsf{T}^+\boldsymbol{\delta}, \quad \text{where} \mathsf{T}^+ = \begin{pmatrix} \mathsf{W}\mathsf{F}_H\mathsf{C}^{-1/2} \\ m^{1/2}\mathsf{P}\mathsf{F}_L\mathsf{C}^{-1/2} \end{pmatrix}. \tag{A12}$$

The $\mathsf{C}^{-1/2}$ operators deconvolve and thus restore the white noise property of the original noise fields. Then, for the high-resolution grid, a high-pass filter is applied (F_H) and the appropriate region is extracted (W). For the low-resolution grid, the low-pass filter is applied (F_L) and the entire domain downsampled ($m^{1/2}\mathsf{P}$, with the $m^{1/2}$ factor serving to preserve the unit variance of the white noise). Further discussion and motivation of this approximate pseudoinverse is given in Appendix B.2.

A.3. Practical Implementation: Generating Zoom Initial Conditions

We next describe how the defining relation (A9) relates to the practical computation of $\boldsymbol{\delta}_H$ at high resolution in the zoom window and $\boldsymbol{\delta}_L$ at low resolution across the full box. For consistency, we should expect that $\boldsymbol{\delta}_H = \mathsf{W}\boldsymbol{\delta}$, i.e., that the high-resolution portion of our final overdensity field is given by extracting the relevant part of $\boldsymbol{\delta}$. However, as previously discussed, a direct computation of $\boldsymbol{\delta}$ is prohibitive, so we are forced to make some approximations. First, let F_{WH} be the high-pass filter defined on the windowed region only. The filter is chosen to remove modes near the fundamental mode of the zoom window, as discussed in Section 2.3. Consequently, we may assume that

$$\mathsf{W}\mathsf{F}_H \approx \mathsf{F}_{WH}\mathsf{W}, \tag{A13}$$

i.e., high-pass filtering and extraction of the zoom window approximately commute. Additionally, we assume

$$\mathsf{F}_L\mathsf{P}^+ \approx \mathsf{P}^+\mathsf{F}_{PL}, \tag{A14}$$

where F_{PL}, is the low-pass filter acting on the pixelized grid. This approximation corresponds to the assumption that the low-pass filter band-limits signals sufficiently far below the Nyquist mode of the pixelized grid, again discussed in Section 2.3. With these two assumptions, the field in the high-resolution region, $\boldsymbol{\delta}_H$, can be approximated as

$$\boldsymbol{\delta}_H = \mathsf{W}\mathsf{T}\boldsymbol{n}_Z \approx \mathsf{F}_{WH}\mathsf{W}\mathsf{C}^{1/2}\mathsf{W}^+\boldsymbol{n}_H + m^{-1/2}\mathsf{W}\mathsf{C}^{1/2}\mathsf{P}^+\mathsf{F}_{PL}\boldsymbol{n}_L. \tag{A15}$$

This expression is not suitable for practical computations, however, because $\mathsf{F}_{WH}\mathsf{W}\mathsf{C}^{1/2}\mathsf{W}^+$ still involves a high-resolution, full-volume convolution. We approximate this operation by applying the covariance matrix evaluated at high resolution but only in the zoom region:

$$\mathsf{F}_{WH}\mathsf{W}\mathsf{C}^{1/2}\mathsf{W}^+ \simeq \mathsf{F}_{WH}\mathsf{C}_H^{1/2}. \tag{A16}$$

Because of the filter F_{WH}, only high-k modes (far above the fundamental frequency of the zoom region) are retained after the convolution; consequently, the approximation should be excellent.

[8] In actual practice, we want to retain the high-frequency modes outside the zoom region rather than cutting them off entirely, so we actually use a modified low-pass filter $\tilde{\mathsf{F}}_L = (\mathbb{I} - \mathsf{W}^+\mathsf{W}) + \mathsf{W}^+\mathsf{W}\mathsf{F}_L$, where F_L is the original Fermi filter. This is equivalent to applying the low-pass filter only in the zoom region, and does not significantly affect how the filters operate.

[9] In general, the covariance of $\boldsymbol{\delta}$ as obtained from $\boldsymbol{n}_H$ and $\boldsymbol{n}_L$ is not precisely C, because outside the zoom region, high-frequency modes contributing to C cannot be represented. The exact covariance C is only obtained in the limit that the zoom region occupies the entire space, $\mathsf{W}^+\mathsf{W} = \mathbb{I}$, and that the F_L band limits the signal in the coarse pixelization, $\mathsf{P}^+\mathsf{P}\mathsf{F}_L = \mathsf{F}_L$.

Similarly, the operation $\mathsf{WC}^{1/2}\mathsf{P}^{+}$ must be replaced: naively, it would involve resampling the volume to high resolution (P^{+}), convolving, then extracting only the high-resolution part (W). Instead, we use the approximation

$$\mathsf{WC}^{1/2}\mathsf{P}^{+} \approx \mathsf{P}_W^{+}\mathsf{W}_P\mathsf{C}_L^{1/2}, \tag{A17}$$

which describes convolving at low resolution ($\mathsf{C}_L^{1/2}$), extracting the zoom region (W_P), and resampling only that region to high resolution (P_W^{+}).

As all of these operations are now tractable (i.e., either they are sampled at low resolution or only encompass the zoom region), they can be efficiently implemented. We thus arrive at the practical estimator used by `genetIC`,

$$\boldsymbol{\delta}_H = \mathsf{F}_{WH}\mathsf{C}_H^{1/2}\boldsymbol{n}_H + m^{-1/2}\mathsf{P}_W^{+}\mathsf{W}_P\mathsf{F}_{PL}\mathsf{C}_L^{1/2}\boldsymbol{n}_L, \tag{A18}$$

which was illustrated in Figure 1. The coarsely pixelated field for the rest of the simulation, $\boldsymbol{\delta}_L$, is obtained by ignoring the irrelevant high-k modes in $\boldsymbol{n}_H$:

$$\boldsymbol{\delta}_L = \mathsf{C}_L^{1/2}\boldsymbol{n}_L, \tag{A19}$$

where C_L is the covariance matrix evaluated at low resolution.

At the time that the overdensity field is computed, we also calculate the velocity field and hence Zel'dovich displacements, which are required for generating particle output. These are obtained using precisely the algorithm above, but using the appropriate covariance matrix, as described in Section 3.2.

A.4. Practical Implementation: Performing Modifications

We are now in a position to derive the algorithm for making modifications to zoom initial conditions. In Section 3.1, we described how, in cases where the uniform high-resolution overdensity $\boldsymbol{\delta}$ is available, modifications are defined via a covector $\boldsymbol{u}$ and a target value d. The aim is to generate $\boldsymbol{\delta}'$, which is statistically as close as possible to $\boldsymbol{\delta}$, but which satisfies

$$\boldsymbol{u}\cdot\boldsymbol{\delta}' = d. \tag{A20}$$

For these constant-resolution vectors, the appropriate linear transformation is given by the Hoffman–Ribak algorithm,

$$\boldsymbol{\delta}' = \boldsymbol{\delta} + \frac{(d - \boldsymbol{u}\cdot\boldsymbol{\delta})\mathsf{C}\boldsymbol{u}}{\boldsymbol{u}\cdot\mathsf{C}\boldsymbol{u}}. \tag{A21}$$

To produce an implementation that works at variable resolution, we first need to find a covector $\boldsymbol{u}_Z$ satisfying

$$\boldsymbol{u}_Z\cdot\boldsymbol{n}_Z = \boldsymbol{u}\cdot\boldsymbol{\delta}, \tag{A22}$$

for the variable-resolution white noise $\boldsymbol{n}_Z$, which generates the overdensity field $\boldsymbol{\delta}$. A general solution can be found by substituting Equation (A9), yielding

$$\boldsymbol{u}_Z = \mathsf{T}^{\dagger}\boldsymbol{u}. \tag{A23}$$

This will, however, be difficult to compute, since it starts from the full high-resolution vector $\boldsymbol{u}$, which by assumption cannot be stored. To simplify, we choose to consider only modifications where the objective is specified within the high-resolution region.[10] For scientific applications, this is naturally the case—the highest resolution is, by construction, centered around the objects of interest.

[10] The resulting modifications will still affect the low-resolution region, as we continue to include the covariance across the entire simulation domain. This is clear, for example, in Figure 5.

Such covectors will satisfy $\mathsf{W}^{+}\mathsf{W}\boldsymbol{u} = \boldsymbol{u}$ because they are zero outside the high-resolution region. This means that we can replace (A23) with a feasible computation,

$$\boldsymbol{u}_Z = (\mathsf{WT})^{\dagger}(\mathsf{W}\boldsymbol{u}), \tag{A24}$$

where we start from $\mathsf{W}\boldsymbol{u}$, which is the $\boldsymbol{u}$ covector calculated only within the high-resolution region. We previously described the approximations used for calculating $\boldsymbol{\delta}_H = \mathsf{WT}\boldsymbol{n}_Z$, and by using the same approximations, we can write $(\mathsf{WT})^{\dagger}$ in a tractable form:

$$(\mathsf{WT})^{\dagger} \approx \begin{pmatrix} \mathsf{F}_H\mathsf{C}_W^{1/2} \\ m^{1/2}\mathsf{F}_{PL}\mathsf{C}_P^{1/2}\mathsf{W}_P^{+}\mathsf{P}_W^{+\dagger} \end{pmatrix}. \tag{A25}$$

We must also transform the covariance matrix C appearing in Equation (A21) to obtain C_Z according to the approximate solution (A9), where T^{+} is specified by Equation (A12). This leads to the result

$$\mathsf{C}_Z = \begin{pmatrix} \mathsf{WF}_H^2\mathsf{W}^{+} & m^{-1/2}\mathsf{WF}_H\mathsf{F}_L\mathsf{P}^{+} \\ m^{1/2}\mathsf{PF}_L\mathsf{F}_H\mathsf{W}^{+} & \mathsf{PF}_L^2\mathsf{F}^{+} \end{pmatrix}. \tag{A26}$$

Once again, this cannot be implemented directly, because it involves operations defined at high resolution over the whole simulation. Applying the same approximations used in obtaining (A25), so that all operations are either performed in the zoom window or at low resolution, we find the appropriate covariance matrix to be

$$\mathsf{C}_Z \approx \begin{pmatrix} \mathsf{F}_{WH}^2 & m^{-1/2}\mathsf{F}_{WH}\mathsf{F}_{WL}\mathsf{W}_P\mathsf{P}_W^{+} \\ m^{1/2}\mathsf{F}_{PL}\mathsf{F}_{PH}\mathsf{P}_W\mathsf{W}_P^{+} & \mathsf{F}_{PL}^2 \end{pmatrix}. \tag{A27}$$

For completeness, we now write the updated Hoffman–Ribak transformation as

$$\boldsymbol{n}_Z' = \boldsymbol{n}_Z + \frac{(d - \boldsymbol{u}_Z\cdot\boldsymbol{n}_Z)\mathsf{C}_Z\boldsymbol{u}_Z}{\boldsymbol{u}_Z\mathsf{C}_Z\boldsymbol{u}_Z}. \tag{A28}$$

When applying multiple modifications, as explained by Roth et al. (2016), we apply Gram–Schmidt orthogonalization and then are able to treat each as independent. The orthogonalization process makes use of the covariance C, which must be replaced by C_Z in the case of zoom simulations. Constraints on the potential or velocity fields can be implemented by replacing all instances of the covariance matrix C with the appropriately reweighted matrix and rederiving C_Z, as described in Section 3.2. Because the case of quadratic modifications (Rey & Pontzen 2018) is linearized and turned into an iterative set of linear modifications, they too can be handled naturally using the updated transformation law (A28).

Appendix B
Technical Details of Approximations

B.1. A Maximum Likelihood Derivation of the Pseudoinverse

In Appendix A.2, we discussed how modifying the initial conditions for zoom simulations requires us to find matrices M_Z that satisfy $\mathsf{MT}\boldsymbol{n}_Z = \mathsf{TM}_Z\boldsymbol{n}_Z$. In general, there is no exact solution to the equation $\mathsf{MT} = \mathsf{TM}_Z$, because T is noninvertible and because careful analysis shows that $\mathsf{T}^{\dagger}\mathsf{T}$ is also noninvertible.

Note that this is a generalized version of the problem considered by Penrose (1956), where a solution is sought for the vector $\boldsymbol{v}$ in the equation $\mathsf{T}\boldsymbol{v} = \boldsymbol{u}$; in that case, $\boldsymbol{v} \simeq \mathsf{T}^{+}\boldsymbol{u}$ is a uniquely motivated approximate solution. We can similarly argue that there is a uniquely motivated choice for obtaining an approximate solution for M_Z. We start by considering the residual error for any candidate M_Z. Since we are describing modifications to $\boldsymbol{\delta}$, which is Gaussian-distributed with covariance C, the error can most naturally be quantified in terms of a χ^2 measure:

$$\chi^2 = \frac{1}{2}(\mathsf{MT}\boldsymbol{n}_Z - \mathsf{TM}_Z\boldsymbol{n}_Z)^\dagger\mathsf{C}^{-1}(\mathsf{MT}\boldsymbol{n}_Z - \mathsf{TM}_Z\boldsymbol{n}_Z). \tag{B1}$$

We now take the expectation value across the ensemble of $\boldsymbol{n}_Z$ (which has unit variance), yielding

$$\langle\chi^2\rangle = \frac{1}{2}\mathrm{Tr}\{\mathsf{C}^{-1}(\mathsf{MT} - \mathsf{TM}_Z)(\mathsf{MT} - \mathsf{TM}_Z)^\dagger\}. \tag{B2}$$

We wish to minimize this expected error—or in other words, maximize the likelihood. We guard against poorly conditioned matrices M_Z by introducing a penalty term $\mathrm{Tr}\{\mathsf{M}_Z\mathsf{M}_Z^\dagger\}$ with a weighting α (this procedure is sometimes known as Tikhonov regularization; see Tikhonov et al. (2013)). The problem is then to find the elements of M_Z that minimize

$$\mathcal{F} \equiv \mathrm{Tr}\{\mathsf{C}^{-1}(\mathsf{MT} - \mathsf{TM}_Z)(\mathsf{MT} - \mathsf{TM}_Z)^\dagger\} + \alpha\,\mathrm{Tr}\{\mathsf{M}_Z\mathsf{M}_Z^\dagger\}. \tag{B3}$$

This is achieved by solving $\partial\mathcal{F}/\partial\mathsf{M}_Z = 0$, leading to the expression

$$\mathsf{M}_Z = (\mathsf{T}^\dagger\mathsf{C}^{-1}\mathsf{T} + \alpha\mathbb{I})^{-1}\mathsf{T}^\dagger\mathsf{C}^{-1}\mathsf{MT}. \tag{B4}$$

We now make a temporary substitution, $\tilde{\mathsf{T}} = \mathsf{C}^{-1/2}\mathsf{T}$, to obtain the simplified expression

$$\mathsf{M}_Z = (\tilde{\mathsf{T}}^\dagger\tilde{\mathsf{T}} + \alpha\mathbb{I})^{-1}\tilde{\mathsf{T}}^\dagger\mathsf{C}^{-1/2}\mathsf{MT}. \tag{B5}$$

Finally, we take the result in the limit that our penalty term is always subdominant, i.e., $\alpha \to 0$. Using the formal definition of the Moore–Penrose pseudoinverse (Penrose 1955) for $\tilde{\mathsf{T}}$, namely

$$\tilde{\mathsf{T}}^+ \equiv \lim_{\alpha\to 0+}(\tilde{\mathsf{T}}^\dagger\tilde{\mathsf{T}} + \alpha\mathbb{I})^{-1}\tilde{\mathsf{T}}^\dagger. \tag{B6}$$

one obtains the solution,

$$\mathsf{M}_Z = \tilde{\mathsf{T}}^+\mathsf{C}^{-1/2}\mathsf{MT} = \mathsf{T}^+\mathsf{MT}, \tag{B7}$$

where we have used $\tilde{\mathsf{T}}^+ = \mathsf{T}^+\mathsf{C}^{1/2}$. This concludes the demonstration that Equation (A11) is the minimum error, i.e., maximum likelihood solution. A similar argument also shows that the maximum likelihood reconstruction of $\boldsymbol{n}_Z$ starting from a given $\boldsymbol{\delta}$ is given by $\boldsymbol{n}_Z = \mathsf{T}^+\boldsymbol{\delta}$.

B.2. Properties of our Approximate Pseudoinverse

In Appendix A.2, we commented that it was not possible to find an exact expression for T^+. We instead motivated the approximate solution (A12). We now provide further information about this approximation and its properties.

First, we note that an exact pseudoinverse satisfies the relations

$$\mathsf{T}^+\mathsf{TT}^+ = \mathsf{T}^+ \quad\text{and}\quad \mathsf{TT}^+\mathsf{T} = \mathsf{T}, \tag{B8}$$

and hence

$$(\mathsf{TT}^+)^2 = \mathsf{TT}^+ \quad\text{and}\quad (\mathsf{T}^+\mathsf{T})^2 = \mathsf{T}^+\mathsf{T}. \tag{B9}$$

This result shows that TT^+ and $\mathsf{T}^+\mathsf{T}$ should both act as projection matrices. The projections can be interpreted as follows: $\mathsf{TT}^+\boldsymbol{\delta}$ takes a general overdensity field $\boldsymbol{\delta}$ and projects out those modes that cannot be represented in our compressed scheme (i.e., high-frequency modes lying outside the zoom region). Conversely, $\mathsf{T}^+\mathsf{T}\boldsymbol{n}_Z$ takes any realization of the white noise $\boldsymbol{n}_Z$ and projects out those modes that are not used in constructing the physical overdensity field $\boldsymbol{\delta}$.

One way to test our approximation for T^+ is to check how closely the projection properties (B9) are adhered to. To start, we show that both requirements are exact in an artificial limit where the zoom region covers the entire volume; in this case, the `genetIC` algorithm continues to split information into high-frequency and low-frequency components, but without any compression. In other words, no pixel downsampling takes place and the high-frequency components are retained across the entire simulation. We thus take $\mathsf{W} = \mathsf{P} = \mathbb{I}$ and $m = 1$, finding that

$$\mathsf{T}^+\mathsf{T} = \begin{pmatrix} \mathsf{F}_H^2 & \mathsf{F}_H\mathsf{F}_L \\ \mathsf{F}_L\mathsf{F}_H & \mathsf{F}_L^2 \end{pmatrix} \quad\text{and}\quad \mathsf{TT}^+ = \mathbb{I}. \tag{B10}$$

The latter follows from the defining relation between the low-pass and high-pass filters, $\mathsf{F}_L^2 + \mathsf{F}_H^2 = \mathbb{I}$, previously stated in Equation (5). From these results, the requirements (B9) follow immediately, confirming that our claimed pseudoinverse is exactly correct.

Let us now consider the more realistic case, where W and P are restored, i.e., high-frequency information is retained only within the zoom region. We now find our approximate pseudoinverse yields

$$\mathsf{TT}^+ = \mathsf{C}^{1/2}(\mathsf{F}_H\mathsf{W}^+\mathsf{WF}_H + \mathsf{F}_L\mathsf{P}^+\mathsf{PF}_L)\mathsf{C}^{-1/2}. \tag{B11}$$

By using the filter relation (5) and factorizing the resulting terms, we find

$$\begin{aligned}(\mathsf{TT}^+)^2 = \mathsf{TT}^+ + \mathsf{C}^{1/2}(\mathsf{F}_H\mathsf{W}^+\mathsf{WF}_L - \mathsf{F}_L\mathsf{P}^+\mathsf{PF}_H)\\ \times(\mathsf{F}_L\mathsf{W}^+\mathsf{WF}_H - \mathsf{F}_H\mathsf{P}^+\mathsf{PF}_L)\mathsf{C}^{-1/2}.\end{aligned} \tag{B12}$$

To obtain an exact projection, it is therefore sufficient that

$$\mathsf{F}_L\mathsf{W}^+\mathsf{WF}_H - \mathsf{F}_H\mathsf{P}^+\mathsf{PF}_L = 0. \tag{B13}$$

Although this equation will not hold in general, there are a number of reasons why it holds to sufficient accuracy for our purposes. First, it involves products of the low-pass and high-pass filters, which when composed produce a narrow bandpass effect. Thus, only a small fraction of modes will be affected by the approximation.

More technically, we can study the properties of overdensity fields $\boldsymbol{\delta}$ for which the operator does obey the desired relation $(\mathsf{TT}^+)^2\boldsymbol{\delta} = (\mathsf{TT}^+)\boldsymbol{\delta}$. By expanding, one finds that $\boldsymbol{\delta}$ must satisfy

$$\mathsf{P}^+\mathsf{PF}_L\mathsf{C}^{-1/2}\boldsymbol{\delta} = \mathsf{F}_L\mathsf{C}^{-1/2}\boldsymbol{\delta}, \tag{B14}$$

$$\mathsf{W}^+\mathsf{WF}_H\mathsf{C}^{-1/2}\boldsymbol{\delta} = \mathsf{F}_H\mathsf{C}^{-1/2}\boldsymbol{\delta}. \tag{B15}$$

The first of these relations, (B14), is always approximately satisfied because, by assumption, $\mathsf{P}^+\mathsf{PF}_L \simeq \mathsf{F}_L$; that is, the entire point of the low-pass filter is to band-limit such that the coarse pixelization becomes irrelevant. The second relation, Equation (B15), states that we must start with overdensity fields that have high-frequency information only within the zoom region. These are precisely the properties we would expect of fields that can be represented accurately in zoom initial conditions.

The operator $\mathsf{T}^{+}\mathsf{T}$, on the other hand, acts on $\boldsymbol{n}_Z$ and takes the form

$$\mathsf{T}^{+}\mathsf{T} = \begin{pmatrix} \mathsf{W}\mathsf{F}_H^2\mathsf{W}^{+} & m^{-1/2}\mathsf{W}\mathsf{F}_H\mathsf{F}_L\mathsf{P}^{+} \\ m^{1/2}\mathsf{P}\mathsf{F}_L\mathsf{F}_H\mathsf{W}^{+} & \mathsf{P}\mathsf{F}_L^2\mathsf{P}^{+} \end{pmatrix}. \tag{B16}$$

We find that the conditions for this to be a projection operator are the same as for $\mathsf{T}\mathsf{T}^{+}$, i.e., they are given by Equation (B13). Thus, for fields that are of interest to `genetIC`, we expect the approximation to be excellent, as borne out by the tests in Section 4.

ORCID iDs

Stephen Stopyra https://orcid.org/0000-0001-8620-4079
Andrew Pontzen https://orcid.org/0000-0001-9546-3849
Hiranya Peiris https://orcid.org/0000-0002-2519-584X
Martin P. Rey https://orcid.org/0000-0002-1515-995X

References

Agertz, O., Pontzen, A., Read, J. I., et al. 2020, MNRAS, 491, 1656
Anderson, L., Pontzen, A., Font-Ribera, A., et al. 2019, ApJ, 871, 144
Angulo, R. E., & Pontzen, A. 2016, MNRAS, 462, L1
Bertschinger, E. 2001, ApJS, 137, 1
Buchert, T. 1993, A&A, 267, L51
Buchert, T., & Ehlers, J. 1993, MNRAS, 264, 375
Daubechies, I. 1992, Ten Lectures on Wavelets, Vol. 61 (Philadelphia, PA: Society for Industrial and Applied Mathematics)
Frigo, M. 1999, in Proc. ACM SIGPLAN 1999 Conf. on Programming Language Design and Implementation, ed. B. G. Ryder, B. G. Zorn, & A. M. Berman (New York: Association for Computing Machinery), 169
Gough, B. 2009, GNU Scientific Library Reference Manual (3rd ed.; Network Theory Ltd)
Hahn, O., & Abel, T. 2011, MNRAS, 415, 2101
Hoffman, Y., & Ribak, E. 1991, ApJL, 380, L5
Jenkins, A. 2013, MNRAS, 434, 2094
Jenkins, A., & Booth, S. 2013, arXiv:1306.5771
Katz, N., Quinn, T., Bertschinger, E., & Gelb, J. M. 1994, MNRAS, 270, L71
Kim, J.-h., Agertz, O., Teyssier, R., et al. 2016, ApJ, 833, 202
Lekien, F., & Marsden, J. 2005, IJNME, 63, 455
Lyth, D. H., & Stewart, E. D. 1990, ApJ, 361, 343
Navarro, J. F., & White, S. D. 1994, MNRAS, 267, 401
Orban, C. 2013, JCAP, 2013, 032
Peebles, P. J. E. 1980, The Large-scale Structure of the Universe (Princeton, NJ: Princeton Univ. Press)
Pen, U.-L. 1997, ApJL, 490, L127
Penrose, R. 1955, PCPS, 51, 406
Penrose, R. 1956, PCPS, 52, 17
Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2016, A&A, 594, A13
Pontzen, A., Slosar, A. c. v., Roth, N., & Peiris, H. V. 2016, PhRvD, 93, 103519
Pontzen, A., Tremmel, M., Roth, N., et al. 2017, MNRAS, 465, 547
Prunet, S., Pichon, C., Aubert, D., et al. 2008, ApJS, 178, 179
Rey, M. P., & Pontzen, A. 2018, MNRAS, 474, 45
Rey, M. P., Pontzen, A., Agertz, O., et al. 2019a, ApJL, 886, L3
Rey, M. P., Pontzen, A., & Saintonge, A. 2019b, MNRAS, 485, 1906
Roth, N., Pontzen, A., & Peiris, H. V. 2016, MNRAS, 455, 974
Sirko, E. 2005, ApJ, 634, 728
Springel, V. 2005, MMNRAS, 364, 1105
Tikhonov, A. N., Goncharsky, A., Stepanov, V., & Yagola, A. G. 2013, Numerical Methods for the Solution of Ill-posed Problems (New York: Springer Science & Business Media)
Tormen, G., Bouchet, F. R., & White, S. D. 1997, MNRAS, 286, 865
van de Weygaert, R., & Bertschinger, E. 1996, MNRAS, 281, 84
Villaescusa-Navarro, F., Naess, S., Genel, S., et al. 2018, ApJ, 867, 137
Zel'dovich, Ya. B. 1970, A&A, 5, 84