INITIAL CONDITIONS FOR LARGE COSMOLOGICAL SIMULATIONS

S. Prunet, C. Pichon, D. Aubert, D. Pogosyan, R. Teissier, and S. Gottloeber

Received 2007 December 19; accepted 2008 May 20

ABSTRACT

This technical paper describes a software package that was designed to produce initial conditions for large cosmological simulations in the context of the Horizon collaboration. These tools generalize E. Bertschinger’s Grafic1 software to distributed parallel architectures and offer a flexible alternative to the Grafic2 software for “zoom” initial conditions, at the price of large cumulated CPU and memory usage. The codes have been validated up to resolutions of 4096^3 and were used to generate the initial conditions of large hydrodynamical and dark matter simulations. They also provide means to generate constrained realizations for the purpose of generating initial conditions compatible with, for example, the local group or the SDSS catalog.

Subject headings: large-scale structure of universe — methods: numerical

Online material: color figures

1. INTRODUCTION

Numerical simulations have proved to be valuable tools in the field of cosmology and galaxy formation. They provide a means to test theoretical assumptions, predict the properties of large-scale structure (and galaxies within), and give access to synthetic observations without sacrificing the whole complexity that arises from nonlinearities. Thanks to recent progresses in numerical techniques and available hardware, numerical cosmology has become one of the most important (and CPU consuming) fields among the scientific topics that require high-performance computing. Over the last few years, a series of large simulations have been produced by, among others, Cen & Ostriker (2000), the Virgo consortium (Frenk et al. 2000) Millennium simulation (Springel et al. 2005), Weinberg et al. (2002), and the Gasoline team (Wadsley et al. 2004). Following the same route, the purpose of the Horizon Project is to federate numerical simulations activities within the French community on topics such as large-scale structure formation in a cosmological framework, the formation of galaxies, and the prediction of its observational signatures. The collaboration studies the influence on the predictions of the resolution, the numerical codes, the self-consistent treatment of the baryons and of the physics included.

These investigations are performed on initial conditions (ICs) that share the same phases, and their production is described in the current paper. The large Horizon ICs involve two boxes of 50 and 2000 Mpc h^{-1} comoving size with, respectively, 1024^3 and 4096^3 initial resolution elements (particles or grid points), following a ΛCDM concordance cosmology. They were generated from an existing set of initial conditions created for the MareNostrum simulation (Gottlöber & Yepes 2007): they share the same phases but with different box sizes and resolutions. The (50 h^{-1} Mpc, 1024^3) ICs were used as inputs to the AMR code RAMSES (Teissier 2002) in a simulation that included dark matter dynamics, hydrodynamics, star formation, metal enrichment of the gas, and feedback. This simulation directly compares to the MareNostrum simulation in terms of cosmology and physics, and we refer to it as the Horizon-MareNostrum simulation (Ocvirk et al. 2008). The (2000 h^{-1} Mpc, 4096^3) ICs served as a starting point for the Horizon-III simulation (Teissier et al. 2007; see also footnote 7). Horizon is a pure dark matter simulation and assumes a cosmology constrained by the WMAP three-year results. It is currently used to investigate the full-sky gravitational lensing signal that could be observed by the Dark Universe Explorer (hence the 411).

The paper is organized as follows: first, we briefly explain the principle of the ICs’ generation. Then we describe how the phases were extracted from the MareNostrum ICs in order to make the Horizon ICs consistent with this reference. The features of codes used to generate and process the different Horizon ICs are as follows:

mpgrafic.—IC generation with optional low-frequency constraints
constrfield.—Low-frequency IC generation with pointlike constraints.
degraf.—Low-pass filtering and resampling of ICs.
powergrid.—IC empirical power spectrum estimation.
splitgrafic.—Estimation of matter density on a grid from a set of particle positions, and Peano-Hilbert domain decomposition.

Finally we illustrate how these codes were implemented on the two Horizon simulations.

2. RANDOM FIELD FOR COSMOLOGICAL INITIAL CONDITIONS

2.1. Grid-based Initial Conditions

For completeness, we quickly review the principle of IC generation. Most of the following has been strongly inspired from articles by Pen (1997) and Bertschinger (2001). Let us consider the initial three-dimensional (3D) Gaussian random field δ(x), representing the density or the displacements, and let us define its Fourier transform δ(k). If we consider zero-mean fields, they
are completely defined by their correlation function or, equivalently, by their power spectra, \( P(k) \):

\[
P(k) \delta_2(k - k') = \langle \delta(k) \delta(k') \rangle.
\]

(1)

All the statistical information in a Gaussian homogeneous and isotropic realization is contained in this quantity, and the difficulty of generating initial conditions resides in obtaining a field that has the correct power spectrum. We chose to follow the convolution-based method described by, for example, Pen (1997) and define the correlation kernel in Fourier space as

\[
A(k) = \sqrt{P(k)}.
\]

(2)

To reproduce the correlation function accurately one may need to first convolve the power spectrum with the window that describes the simulation box, as advocated by Pen (1997). The influence of the box size on rms density in spheres of given radius (which are relevant for mass function estimates of collapsed objects) is negligible for sphere radii much smaller than the box length, even for simulations designed to study galaxy cluster scales.

Then the ICs generation is a two-step procedure. First a normal, uncorrelated random field of unit variance is generated in position space. This white noise \( n_1(x) \) has a constant power spectrum; that is,

\[
\langle |n_1(k)|^2 \rangle = 1.
\]

(3)

The second step involves convolving the white noise with the correlation kernel in order to obtain the initial fluctuation field,

\[
\delta(x) = n_1(x) * A(x),
\]

(4)

or in Fourier space

\[
\delta(k) = n_1(k) A(k).
\]

(5)

It can be easily seen that \( \langle |\delta(k)|^2 \rangle = P(k) \), and the initial field automatically has the correct power spectrum.

One of the main virtues of the method is the possibility of using the same white noise for different power spectra. In other words, it explicitly decouples the phases (which contain the specificities of a given realization in terms of relative positions) from the amplitudes of the fluctuations (corresponding to one’s favorite choice of cosmological model). A change in the physics or in the box size results in a change of the convolution kernel, but the underlying structure of the field will remain globally the same for a given white noise realization. Conversely, equation (5) implies that the initial phases can be recovered from a set of ICs, provided that the convolution operation can be inverted. In other words, it is possible to generate a new set of ICs from an old one (see § 4), and such a set would share the same overall structures with, for example, a modified cosmology or box size.

2.2. Grid-based Versus Particle-based Initial Conditions

In numerical simulations, the dark matter distribution is almost exclusively described in terms of particles, and this discretized description is also applied to the gas in SPH-like hydrodynamical codes. Consequently, dealing with “particle-type” data is the most frequent case, while the current procedure naturally deals with densities and velocities sampled on a grid. This can be easily tackled by recalling the density-velocity relation that is valid in the linear regime:

\[
\frac{1}{aH} \nabla_x u = -f(\Omega_m, \Omega_\Lambda) \delta(x).
\]

(6)

Here \( u \) stands for the comoving peculiar velocity, \( x \) for the comoving position, \( H \) for the Hubble constant, \( a \) for the scale factor, and \( f \) is defined as the logarithmic time derivative of the growth factor:

\[
f(\Omega_m, \Omega_\Lambda) \equiv \frac{d \log D^+}{d \log a}.
\]

(7)

Functional fits for \( f \) can be found in the literature (e.g., Lahav et al. 1991) or be directly computed for a given cosmology. Hence, assuming that particles were displaced from a regular grid and knowing their velocities, the initial density field can be directly recovered from equation (6). One can see that the (Eulerian) positions of particles are not directly involved in this procedure; however, their Lagrangian coordinates are used to remap the particle velocities to grid cells.

3. GRID-BASED ICs: TECHNICAL IMPLEMENTATION

Once the “phases” (the white noise) are chosen on a grid of a given size, it is possible to use them to generate initial density and velocity realizations with the desired cosmology and power spectrum, at resolutions that can differ from the initial white noise realization.

According to linear theory, which applies for these initial conditions, density and velocity divergence are related through equation (6), so that a single white noise realization determines both density and velocities on a grid of equal size (see, e.g., Bertschinger 2001). A first numerical implementation of this algorithm was made by E. Bertschinger in the package Grafic1. Here we extend his code using the message passing interface (MPI) library to deal with large simulation cubes on distributed memory platforms. Another implementation of MPI-based initial condition generator in the context of the GRACOS code is described in Shirokov (2007). We also develop a few tools (low-pass filtering and resampling, power spectrum estimation, estimating matter density on a grid from a set of particle positions) that work as well in parallel environments. We describe these tools and their usage in the following subsections.

3.1. Mggrafic: A Parallel Version of Grafic1

Generating initial conditions for cosmological simulations on a grid from an initial white noise realization is theoretically quite simple, as it involves a straightforward implementation of equations (5) and (6) in Fourier space. Indeed, the main issues in the Grafic1 code involve getting the normalization right (in terms of \( \sigma_8 \) or \( Q_{\text{rms}} \), which are input parameters), and therefore in the cosmology routines.

3.1.1. Description

Difficulties of a more technical nature appear as the size of the desired cubes (and/or the number of particles) grow and that

---

7 We could have directly generated the real and imaginary part of each \( \delta(k) \) following a \( \mathcal{N}(0, 1/\sqrt{2}) \) law, saving the cost of an extra Fourier transform, but we chose to remain compatible with the Grafic code.

8 Actually, the validity of the linear theory is enforced by choosing the starting redshift in such a way that the resulting variance of the discrete density field is significantly less than unity.

9 See also http://www.gracos.org.
a single cube does not fit into a computer’s memory. An elegant answer to this problem, in the context of multiresolution (“zoom”) initial conditions was designed by Bertschinger (2001) and implemented in the Grafic2 package. This solution involves generating a low-resolution cube first, and successively adding higher frequencies in nested subregions, constrained by the phases of the already existing low-frequency modes. Strictly speaking, the exact solution to this problem is (naively) as costly as the direct generation of the full cube at the highest resolution, but approximate, less costly solutions based on anti-aliasing kernels can be designed. This is precisely what is done in the Grafic2 package.

The main advantage of this solution is to produce multiresolution initial conditions with a very reasonable memory usage, but it also has drawbacks, namely its complexity, and its built-in restrictions on the nested cubes structure of a given maximum size. Given the growing size of computer clusters, our “brute force” approach to the problem based on parallelism becomes possible, and it is also in some ways more flexible. First of all, it allows for a direct generation of global initial conditions for very large cosmological simulations. Second, it also allows, together with associated tools for low-pass filtering and resampling, to create multiresolution simulations of a more general structure by simply extracting the desired subregions from the series of “downgraded” cubes (obtained by low-pass filtering and resampling of the initial large high-resolution cube).

There are two issues that arise when implementing a parallel (MPI-based) version of the Grafic1 software. The first involves performing efficient 3D fast Fourier transforms (FFTs); this difficulty is solved by using the parallel version of the FFTW\textsuperscript{10} library, which uses a slab domain decomposition. The second difficulty lies in the input/output. Indeed, we decided to keep the binary (Fortran) structure of the files in the Grafic1 format. In a parallel environment where each MPI process is responsible for one chunk of data, this lead us to write part of the I/O subroutines in C using reentrant read/write routines, wrapped in FORTRAN90 to be callable from the main program.

Finally, apart from the parallelism of mpgrafic, we have added a few new features to the original Grafic1 code. An implementation of the matter power spectrum with baryon oscillations was introduced, as described in Eisenstein & Hu (1998). Second, the possibility of constraining the low-frequency phases of the density and velocity realizations was implemented, with the input of a given white noise cube of lower resolution. This allowed us to use the same set of initial condition phases for cosmological simulations at different resolutions. Last, we have implemented the possibility of constraining the value of the density or velocity field values, as well as their gradients and hessians at a chosen set of positions. We return in more detail to this last point in a following section, as it is a nontrivial extension of the code.

3.1.2. Code Installation and Parameter File Structure

A prerequisite to use mpgrafic is to have (of course) a valid MPI library installed, including a FORTRAN90 compiler. A second prerequisite is to have the fftw-2.1.5 library installed, with the --enable-mpi --enable-type-prefix options at the configure step. The first option builds the static and shared FFTW MPI-based libraries, the second is for the type (float or double) naming scheme of the libraries.

Note that the default build of FFTW corresponds to double precision, whereas the default type in mpgrafic is single precision. To change to single precision realizations, you need to make the single precision FFTW MPI libraries by adding the --enable-float at configure time. To compile mpgrafic in double precision mode, you need to configure with the --enable-double keyword.

The code usage has been kept as close as possible to the Grafic1 code interface, except of course for the few additional options to the code. In Table 2 in the Appendix we show an example of an mpgrafic parameter file.

Compared to the original Grafic1 parameter files, the only differences lie in the possibility of an input power spectrum with baryonic oscillations (Eisenstein & Hu 1998), and in the optional input of a small noise file to constrain the large-scale phases. Otherwise, the code is called with mpirun -np <#proc> mpgrafic < parameter_file. Like Grafici, it produces seven data cubes (one density file, three dark matter velocity files, and three baryon velocity files). In the file grafic1.inc, the offsets of the velocity fields is controlled by the parameters offvelb, offvelc. The redshift of the realizations is controlled by the variance of the density field on the grid, as specified by sigstart. Finally, the size of the cubes is controlled by the parameters np1, np2, and np3, which can take different values and are set in grafic1.inc. Note that when these parameters are changed the code needs recocompilation.

3.1.3. Illustration on a Small Example, Power Spectrum Estimate

In Figure 1 we show a slice of a density file realization with the parameter file displayed in Table 2 below, but without large-scale constraints, and for np1 = np2 = np3 = 256. In the same figure, we show the density files of linear grid size 128 and 64 obtained from this realization, by low-pass filtering and subsampling. This operation has been done using the utility called degraf, that makes use of the FFTW parallel Fourier transform routines. Written in FORTRAN90, it takes as input a collection of files in Grafic1 format, as well as some parameters in a namelist file. These parameters include the list of the input file names, the target resolution of the output files, and an optional shift vector allowing a global translation of the output files (with periodic initial conditions).

Finally, another utility, powergrid, uses the FFTW parallel Fourier transform routines to compute the periodogram estimate of a density field power spectrum. It allows correction for nearest grid point (NGP) or cloud in cell (CIC, linear) interpolation of a particle set to the computation grid. Note that this remapping of a discrete density field given by particles onto grid cells leads not only to a smoothing of the grid Fourier modes (this can be corrected by the code) but also to some power aliasing, which cannot be corrected unless prior knowledge of the density power spectrum is available. These points are illustrated in § 4, for the Horizon simulations. Of course, none of these problems appears if one is only interested in computing the periodogram of the IC grid-sampled density fields. Figure 2 shows the theoretical power spectrum corresponding to the density field realization shown in Figure 1, together with its periodogram estimate, as well as the periodograms of its low-passed, resampled versions.

3.2. Constrained Initial Conditions

Since mpgrafic opens the opportunity of generating ICs that are consistent with a given low-frequency cube, it is interesting to build such a cube of phases so that the overall cube satisfies (low-frequency) pointlike constraints on a given set of points. These constraints fix the mean value of the density or velocity field, as well as their gradients and hessians, computed at a chosen set of positions, for a given smoothing length $R_p$ (see eq. [11] below). Once such a low-frequency cube has been generated, it
is then whitened and set as an input to mpgrafic, which is then used to add small-scale power and to resample the field according to the new Nyquist frequency.  

This procedures allows, for instance, the generation of initial conditions that are consistent with, say, a given merging event or the structure of the Local Group, the large-scale structures derived from large surveys such as the SDSS (Adelman-McCarthy et al. 2008), the 2dF (Percival et al. 2001), 2Mass (Jarrett et al. 2003), and the Local Group (Mohayaee & Tully 2005).

11 In principle, adding small-scale power with mpgrafic, while keeping the same low-frequency phases, violates the pointlike constraints imposed by constrfield. However, if the smoothing kernel \( W(kR_p) \) and its associated smoothing length are chosen in such a way that it cuts all modes with wave-vectors above the initial Nyquist frequency, the constraints are not violated by adding small-scale power.

The ensemble of unconstrained Gaussian random fields is defined by all possible realizations of the field values \( \delta(\mathbf{x}) \), or their Fourier amplitudes \( \delta(k) \), for a given power spectrum. If we impose the constraints on the field, the statistical ensemble narrows down to a subset of realizations, those that have the constraints satisfied. In particular, for a discrete representation of the field on a grid of size \( N^3 \), this means that not all \( N^3 \) values of \( \delta(\mathbf{x}) \) are real degrees of freedom. Averaging over the constrained ensemble \( \langle \ldots \rangle_c \) makes both the mean \( \langle \delta(\mathbf{x}) \rangle_c \) and the variance dependent on position \( \mathbf{x} \).

We deal with linear constraints each of which, in general, sets a linear functional relation \( V_a(\delta(\mathbf{x})) \) (we use Latin letters to index the constraints, where each constraint is defined by a grid position \( \mathbf{x}_a \) and a constraint operator \( Y_a \), see below) between field values to a given value, \( V_a(\delta(\mathbf{x})) = \bar{V}_a \).

Alternatively, we can take a point of view that such a restricted ensemble defines a new (constrained) random field \( \delta_c(\mathbf{x}) \), which is still Gaussian (due to the linearity of imposed relations), but is statistically inhomogeneous. Under these conditions, the well-known way to construct a constrained field from unconstrained realizations is

\[
\delta_c(\mathbf{x}) = \delta(\mathbf{x}) + \sum_{ab} \langle \delta(\mathbf{x}) V_a[\delta] \rangle \langle V_a[\delta] V_b[\delta] \rangle^{-1} \left( \bar{V}_b - V_b[\delta] \right)
\]

(see, e.g., Bardeen et al. 1986; Hoffman & Ribak 1991). In this expression, the random quantities on the right-hand side are unconstrained \( \delta(\mathbf{x}) \). Here \( \bar{V}_b \) is a numerical value of the constraint \( b \), while \( \langle \delta(\mathbf{x}) V_a[\delta] \rangle \) is the covariance between the field and a constraint, and \( C_{ab} = \langle V_a[\delta] V_b[\delta] \rangle \) is the covariance matrix between the constraint functionals. This recipe reproduces the mean (note, the averaging is taken over all unconstrained realizations)

\[
\langle \delta_c(\mathbf{x}) \rangle = \langle \delta(\mathbf{x}) V_a \rangle \langle V_a V_b \rangle^{-1} \bar{V}_b,
\]

and the correlation function

\[
\langle \delta_c(\mathbf{x}) \delta_c(\mathbf{x}') \rangle = \langle \delta(\mathbf{x}) \delta(\mathbf{x}') \rangle V_a V_a^{-1} \langle \delta(\mathbf{x}') \rangle V_b + \langle \delta(\mathbf{x}) \rangle \langle \delta(\mathbf{x}') \rangle,
\]

which define all statistical properties for the Gaussian case.
In cosmology the primary interest is to define constraints that describe the physical properties of a patch (see, e.g., Bond & Myers 1996; van de Weygaert 1996) of the density field density, density derivatives, and shear flow, averaged over the volume of the patch. Such constraints can be represented in Fourier space as

$$V_p[\delta] = \int d^3 k \delta(k) Y_p(k) W(k R_p) e^{-ikx_\delta},$$

where $x_\delta$ is the position of the patch, $W(k R_p)$ is a averaging filter over the patch size $R_p$, and $Y_p(k)$ is the Fourier representation of the operator that specifies the constraint functional. In particular, we use the constraint types given in Table 1. Using constrained field formalism, Bond et al. (1996) have demonstrated that the observed filamentary cosmic web of matter distribution in the universe can be understood as dynamical enhancement of the geometrical properties of initial density field. The web is largely defined by the position and primordial tidal fields of rare events in the medium, such as precursors of large galaxies at high redshifts or clusters of galaxies at present time, with the strongest filaments between nearby clusters whose tidal tensors are nearly aligned.

The code constrfield implements the same cosmology as mpgrafic (including baryon wiggles) and offers the possibility of whitening\(^{12}\) the resulting field in order to feed it to mpgrafic as a low-frequency input. An example of this procedure is illustrated in Figure 3.

### 3.3. Splitting the ICs

Starting a parallel computation requires the initial conditions to be dispatched over all the computing processes. Two alternatives exist to perform this operation. The first one involves having the initial conditions to be read by a master process and the data to be broadcast to all the other processes, and then keep or reject the broadcaster data according to the topology of the computation’s domain split. While simple to implement, this option is difficult to use in practice since broadcasting over a large number of processes can be technically problematic and time consuming.

We present an alternative option that involves having the processes upload their own set of data only. Because it is wasteful for each process to parse the whole set of initial conditions to get the relevant subset of data, this option implies that initial conditions are presplit according to the domain decomposition strategy of the integrator. This splitting is both a domain boundary assignment, a procedure for distributing particles among the processes, and a per process file dump.

It results in a faster procedure, since each process reads its own set of data, instead of having one process reading the full initial conditions. A possible drawback lies in the fact that a splitting is defined a priori; changing the number of processes dedicated to the computation therefore requires the production of a whole new set of split initial conditions. However, splitting can be performed on a single process, prior to any parallel computation, and exhibits a negligible cost in terms of CPU time. Such a procedure can thus be applied an arbitrary number of times at almost no expense.

Horizon simulations were started from split initial conditions, where each process uploaded its own set of data. The splitting scheme followed the domain decomposition’s strategy of the cosmological calculations performed with RAMSES. It relies on a 3D Peano-Hilbert space filling curve (Salmon & Warren 1997; MacNeice et al. 2000), which provides a complete mapping between the 3D position of a grid point and a 1D coordinate on this curve. A two-dimensional (2D) example of such a curve is shown in Figure 4. By using this piecewise linear representation of the computation domain, each process is being given a

12 By “whitening,” we mean the operation that transforms an unconstrained field into a white noise, i.e., a collection of independent identically distributed random variables $N(0, 1)$. Note that the presence of constraints breaks the independence of the grid cells even after “whitening” (see eq. [10]).
continuous portion of this curve and load balancing is achieved by “sliding” the limits of the local data sets along the space-filling curve. In particular, the initial conditions are by construction well balanced; therefore, the splitting among all the processes involves a set of even subdivisions of the Peano-Hilbert curve. A set of successive \((q_1, q_2, \ldots, q_n)\) corresponds to a single process.

Note that all subdomains are simply connected, that is, within there are no isolated subregions owned by a process in the middle of another region owned by another process. For instance, in 3D, if the curve is split in 2 sections, each section fits in a 3D rectangle of different sizes and orientations (see Fig. 5). Moreover, if the curve is split in 8 parts, all the sections fit in a cube of the same volume. This type of domain decomposition is known to be most efficient if we consider the ensemble of all the refined grids configurations. It may be surpassed by other strategies (e.g., layer splitting, angular splitting) on specific situations.

Here we implemented a fast and simple algorithm that performs the splitting of initial conditions according to the Peano-Hilbert domain decomposition. It relies on a plane-by-plane investigation of the data cube, which limits the memory consumption. Let \(n^3\) be the number of cells in the initial conditions data cube, \(F(i, j, k)\) the value of the 3D field at grid indices \(1 \leq i, j, k \leq n\), and \(f\) a 2D slice of \(F\) at index \(k\). We assume that the number of processes \(nproc\) is a power of 2 (even though this constraint can be lifted as explained below); it ensures that each process domain is a 3D rectangle. Consequently the extent of the subdomain corresponding to a process \(p\) is given by two triplets \((i_m, j_m, k_m)\) and \((i_M, j_M, k_M)\), which correspond to the two extreme corners of the rectangle. The algorithm described below runs on one process only, and involves two distinct steps:

1. **Do** \(k = 1, n\): loop over data planes.
2. Read \(F(1 : n, 1 : n, k)\): the 2D data is uploaded.

**Initialization:**

\[
\text{process}(1 : nproc) = \text{true},
\]

\[
(i_m, j_m, k_m)^p = -1, (i_M, j_M, k_M)^p = -1, \forall p \in [1 : nproc].
\]

**Do** \(j = 1, n, i = 1, n\). Peano-Curve mapping: \((i, j, k) \rightarrow q \rightarrow p\).

- process \(p\) is found in the \(k\)-plane: \(\text{process}(p) = \text{true}\).
- If process \(p\) is found for the first time, set the minimum position: if \((i_m, j_m, k_m)^p = -1\), then \((i_m, j_m, k_m)^p = (i, j, k)\).
- Update the maximum position: if \((i, j, k) > (i_M, j_M, k_M)^p\), then \((i_M, j_M, k_M)^p = (i, j, k)\).

**Do** \(p = 1, nproc\).

- If \(\text{process}(p) = \text{false}\), skip: only processes detected in the plane are taken in account.
- Write \(f(i_m^p : i_M^p, j_m^p : j_M^p, k_m^p : k_M^p)\) in the file of the process \(p\).

First, initial conditions fields are parsed plane by plane, and for each plane, the process map is achieved through the space-filling curve mapping. Then for each process found in the current plane, the subset of data is written in the relevant files. Because of the compactness of the subdomains, a significant speedup can be obtained by parsing the \(i\) and \(j\) indexes of the second loop with steps larger than 1. This procedure is safe as long as the step remains smaller than the smallest extent of the subdomains along one direction. We call this step the speedup step. Finally the \(nproc = 2^3\) constraint can be lifted if a set of processes upload the same set of data. Each process would load a small initial condition file and would retain only its subsubdomain within a subdomain. For instance the Horizon 4PI simulation ran on 6144 = 3 \times 2048 processes: the splitting was performed over 2048 subdomains and each subfile was uploaded by three processes. Overall, the algorithm can be quite effective, and as an illustration, the 4096\(^3\) initial conditions fields of this simulation were split in 15 minutes on a single process of the CCRT computing center using a speedup step of 256. The code ran on Itanium2 processors (double-core, but only a single core has been used here) with a 1.6 GHz frequency.

4. APPLICATION TO LARGE HORIZON SIMULATIONS

Let us now illustrate on a couple of large-scale simulations how mpragif was used. We consider in turn a hydrodynamical and a dark matter only simulation.
4.1. Horizon-MareNostrum

The first major application of the mpgrafic code was the generation of ICs for a simulation of a cosmological hydrodynamical simulation of linear size 50 h^{-1} Mpc on a grid of size 1024^3.

For the MareNostrum simulation, we started for technical reasons with external ICs (given as a set of particle velocities), as one of the goals was to compare two n-body plus hydro codes (namely the RAMSES and GADGET2 codes) on similar ICs. The particle velocities were therefore read from external ICs and we performed the derivation of the density field samples on the grid from the particle velocities in Fourier space, using the JMFFT Fourier Transform package. The f factor involved in equation (7) was computed using routines provided in the original Grafic package. Once the velocity divergence and f are known, the initial density field is easily recovered and ready to be used as an input for simulations (especially for the hydrodynamical part of RAMSES), or as a source for a specific set of initial phases.

4.1.1. Power Spectrum Extraction from MareNostrum Initial Conditions

The master equation (eq. [5]) can only be inverted knowing the convolution kernel, that is, the power spectrum P(k). In principle, the knowledge of the cosmology and the included physics should be sufficient to derive P(k) prior to the deconvolution. Let us call PM(k), this theoretical power spectrum constrained only by physics. In practice, this theoretical power spectrum differs from the effective power spectrum used to draw the (external, particle based) ICs. Our goal in this section is to define a power spectrum P_{final}(k) that should accurately represent the ensemble averaged power spectrum of the external ICs [so that (P_{mean}(k)) \approx P_{final}(k)], based on the available theoretical and measured power spectra. P_{final}(k) will then be used in the deconvolution.

Using an inaccurate spectrum to deconvolve equation (5) would lead to a “colored” noise for the initial phases, that is, with spurious characteristic length scales.

Let us illustrate the discrepancy between the theoretical and the measured P(k) by describing the power spectrum of the MareNostrum ICs. The measured P(k) is shown in Figure 6 as triangles. The theoretical P(k) = P_{in}(k) (i.e., the one used to generate this set of ICs) is also shown as a dotted line. Unsurprisingly, the two curves disagree.

At low k, the finite volume of the simulation implies that the empirical power spectrum of large-scale modes has large sampling variance and thus departs from the theoretical curve. The high k discrepancy is of a different nature: clearly the sampling variance is negligible, but now the discreetness of the grid and anisotropy of very high k mode representation play role. P(k) departs significantly from P_{in}(k) as easily seen when zooming on the large k regions, where P(k) lacks power compared to the expected behavior. Therefore, P_{in}(k) cannot be used without corrections to whiten the external IC set. In the following, we rely on the fact that Gaussian initial conditions are statistically characterized by power spectrum only.

The exact set of corrections depends on how the field has been generated. For MareNostrum ICs we must first include a Hanning filter defined in the Fourier space by

$$W_{H}(k) = \cos\left(\frac{\pi k}{2N}\right).$$

Here the Nyquist frequency is given by k_N = 2\pi/L \times N/2, where L = 50 Mpc h^{-1} is the size of the box of the MareNostrum ICs and N stands for the (linear) number of grid elements. Such a filtering is frequently encountered when dealing with initial conditions: because Fourier modes are sampled on a Cartesian grid, the two conditions k < \pi N/2 and (k_x, k_y, k_z) < \pi N/L imply that anisotropies arise on the smallest scales along the diagonals. The Hanning filter damps high-frequency modes and reduces the small-scale contributions and consequently the anisotropies. In Figure 6, we display the P_{H} curve as a dashed line, where

$$P_{H} = W_{H}(k)^2 P_{in}(k).$$

Clearly, P_{H}(k) reproduces well the measured power spectrum (except at high frequencies; see below), with N = 2048, which corresponds to the original resolution of the external ICs prior to

\[\text{Fig. 6.—Power spectrum } P(k) \text{ of the MareNostrum initial conditions. Symbols stand for the measured power spectrum, } P_{\text{measured}}(k), \text{ while the dotted line is for the theoretical power spectrum } P_{\text{in}}(k). \text{ The dashed line, } P_{\text{H}}(k), \text{ is for the theoretical power spectra plus a Hanning filter contribution. The solid line, } P_{\text{final}}(k), \text{ is for our best fit of the power spectrum. The right panel represents a zoomed version of the left one with only the small scales shown.}\]
some (external) degradation procedure. This modification of the spectrum corresponds to the most favorable case in which an analytic expression is known or can be found for the filtering applied on the data.

Second, Figure 6 shows that $P_H(k)$ still lack some power for $k > 40 \, h^{-1} \text{Mpc}$. This feature corresponds to the external degradation procedure: one particle out of eight was provided, out of the original 2048$^3$ particles, resulting in power aliasing at high frequency. This part of the power spectrum was fitted by a smoothed version of the measured power spectrum. We call $P_{\text{final}}(k)$ this final power spectrum that includes the effect of the Hanning Filter and corrects the high-frequency effects due to the degradation:

$$P_{\text{final}}(k) = P_H(k)k \leq 40 \, h^{-1} \text{Mpc}^{-1},$$

$$S[P_{\text{meas}}(k)]k > 40 \, h^{-1} \text{Mpc}^{-1},$$

where $S[X]$ stands for a smoothing operator. By using a smoothed version of the spectrum, we avoid overfitting of the fluctuations in the spectrum, which would artificially reduce the variance at these scales.

To conclude, $P_{\text{final}}(k)$ combines both an analytic expression of the filtered theoretical spectrum at low frequencies and a numerical evaluation at high frequencies, based on the measured power spectrum. We emphasize that these choices are by no means unique but were found to provide phases with the proper spectrum.

4.1.2. Whitening

The “whitening” operation [i.e., getting the white noise $n_1(x)$ from $\delta(x)$ and $P_{\text{final}}(k)$] is then performed by deconvolution as in equation (5). For the Horizon-MareNostrum ICs, the power spectrum of the resulting white noise is shown in Figure 7. The whitened ICs are now ready to be processed into new initial conditions: the whitened ICs serve as low-frequency constraints when generating the refined ICs using mprgafic.

The code mprgafic was then used to generate the ICs of a ΛCDM hydrodynamical “full physics” simulation (with star formation and metals) with a box size of 50 $h^{-1}$ Mpc on a grid of 1024$^3$ using the Horizon reference white noise. The comparison between the input phases and the reproduced phases with mprgafic is illustrated in Figure 8, which shows both simulations at redshift 5.7.

4.2. Horizon-4H

As mentioned earlier, the code mprgafic was also used to generate the ICs of Horizon-4H, a ΛCDM dark matter only...
simulation based on cosmological parameters inferred by the WMAP three-year results, with a box size of 2 h⁻¹ Mpc on a grid of size 4096³ (see Fig. 9). The purpose of this simulation is to investigate full sky weak lensing and baryonic acoustic oscillations. The 70 billion particles were evolved using the particle mesh scheme of the RAMSES code on an adaptively refined grid (AMR) with about 140 billions cells. Each of the 70 billion cells of the base grid was recursively refined up to six additional levels of refinement, reaching a formal resolution of 262,144 cells in each direction (roughly 7 kpc h⁻¹ comoving).

The corresponding power spectrum was measured using powergrid, and is shown in Figure 10. Since the simulation snapshots involves a collection of particles, we had to resample them on a grid using a convolution kernel before estimating the power spectrum. The resulting (analytical) bias in the power spectrum was corrected; however, this resampling procedure leads to some power aliasing close to the Nyquist frequency that cannot be corrected without additional information. Finally, note that the AMR structure of the RAMSES code leaves the opportunity of measuring the power spectrum at frequencies beyond the Nyquist frequency of the 4096³ grid. Such measurements are outside the scope of this paper, and the power spectrum tool powergrid should be viewed primarily as a diagnosis tool. Here, as a check of both the initial condition generation algorithm, including the implementation of the baryon wiggles, a novelty of this implementation, Figure 11 displays the measured baryon wiggles in the z = 0 snapshot, together with the corresponding fit.

5. CONCLUSION

A series of tools to construct and validate initial conditions for large (n > 1024³) cosmological simulations in parallel are presented and illustrated. These tools involve ICs generation with optional constraints, low-pass filtering and resampling, power spectrum estimation, estimation of matter density on a grid from a set of particle positions, and Peano-Hilbert domain decomposition. As illustrated in § 4, they allowed us to produce very large cosmological simulations. From these high-resolution ICs, one can then create at will zoomlike initial conditions and constrained ICs with the help of the resampling tool. Let us emphasize that powergrid provides an alternative route to initial condition generation such as Grafic2. It is more versatile, as it does not impose any relationship between resolution and boundaries for the refined subvolumes. The remaining limitation is the total amount of memory available on distributed architectures. A logical extension of this work will be to generate initial conditions corresponding to the local group. Note finally that with simple amendments, the above mentioned code could be used in the context of vector field generation (magnetic field with a given helicity), or turbulence.

We warmly thank the Barcelona Supercomputing Center and the CCRT staff for their help in producing the Horizon-MareNostrum and the Horizon-4III simulations. We also thank the referee for his careful reading of the manuscript, G. Yepes for his help with the MareNostrum initial conditions, Stephane Colombi, Karim Benabed, Julien Devriendt, Thierry Sousbie for advices, and D. Munro for freely distributing his Yorick programming language and opengl interface (available at http://yorick.sourceforge.net). This work was carried within the framework of the Horizon project: http://www.projet-horizon.fr. All codes described in this paper are available at ftp://ftp.iap.fr/pub/from_users/prunet.
## APPENDIX

**MPgrafics PARAMETER FILE**

In this appendix we present a sample MPgrafics parameter file (Table 2).

### Table 2

**mpgrafic Parameter File Example**

| Parameters | Description |
|------------|-------------|
| 4............... | Power spectrum parameterization: 4 is Eisenstein & Hu |
| 0.24, 0.76, 73.0 | $\Omega_m$, $\Omega_L$, $H_0$ |
| 0.042 | $\Omega_b$ |
| 0.96 | $\sigma_8$ |
| -0.92 | Normalization: $-\sigma_8$ if negative, $\Omega_m$ if positive |
| 0.01, 100.0 | $k_{\text{min}}$ and $k_{\text{max}}$ in $h^{-1}$ Mpc for analytical PS output |
| -50.0 | Box length in $h^{-1}$ Mpc if negative, mesh length in Mpc if positive |
| 1 | Grafic1 mode: no choice here |
| 0 | Grafic1 mode: no choice here |
| 1 | 1. Generate noise file and save it; 2. Read from noise file |
| 1234 | Initial seed (useful if 1 is set above) |
| white-256.dat | Noise file name |
| 1 | 1. Constraint of large scale phases with small noise file; 2. No padding |
| white-128.dat | Small (constraint) noise file name |

---

**REFERENCES**

Adelman-McCarthy, J. K., et al. 2008, ApJS, 175, 297
Bardeen, J. M., Bond, J. R., Kaiser, N., & Szalay, A. S. 1986, ApJ, 304, 15
Bertschinger, E. 2001, ApJS, 137, 1
Bond, J. R., Kofman, L., & Pogosyan, D. 1996, Nature, 380, 603
Bond, J. R., & Myers, S. T. 1996, ApJS, 103, 1
Cen, R., & Ostriker, J. P. 2000, ApJ, 538, 83
Eisenstein, D. J., & Hu, W. 1998, ApJ, 496, 605
Frenk, C. S., et al. 2000, preprint (astro-ph/0007362)
Gottlöber, S., & Yepes, G. 2007, ApJ, 664, 117
Hoffman, Y., & Ribak, E. 1991, ApJ, 380, L5
Jarrett, T. H., Chester, T., Cutri, R., Schneider, S. E., & Huchra, J. P. 2003, AJ, 125, 525
Lahav, O., Lilji, P. B., Primack, J. R., & Rees, M. J. 1991, MNRAS, 251, 128
MacNeice, P., Olson, K. M., Mobarry, C., de Fainchtein, R., & Packer, C. 2000, Comput. Phys. Commun., 126, 330
Mohayaee, R., & Tully, R. B. 2005, ApJ, 635, L113
Ocvirk, P., Pichon, C., & Teyssier, R. 2008, MNRAS, in press
Pen, U.-L. 1997, ApJ, 490, L127
Percival, W. J., et al. 2001, MNRAS, 327, 1297
Salmon, J., & Warren, M. S. 1997, in Proc. 8th SIAM Conference on Parallel Processing for Scientific Computing, ed. M. Heath (Philadelphia: SIAM)
Shirokov, A. 2007, preprint (arXiv: 0711.4655)
Smith, R. E., et al. 2003, MNRAS, 341, 1311
Springel, V., et al. 2005, Nature, 435, 629
Teyssier, R., et al. 2002, A&A, 385, 337
Teyssier, R., et al. 2007, A&A, submitted
van de Weygaert, R. 1996, in ASP Conf. Ser. 94, Mapping, Measuring, and Modelling the Universe, ed. P. Coles, V. Martinez, & M.-J. Pons-Borderia (San Francisco: ASP), 49
Wadsley, J. W., Stadel, J., & Quinn, T. 2004, NewA, 9, 137
Weinberg, D. H., Hernquist, L., & Katz, N. 2002, ApJ, 571, 15