Perfectly parallel cosmological simulations using spatial comoving Lagrangian acceleration

F. Leclercq, B. Faure, G. Lavaux, B. D. Wandelt, A.H. Jaffe, A.F. Heavens, W.J. Percival, Camille Noûs

A&A 639, A91 (2020)
https://doi.org/10.1051/0004-6361/202037995

Astronomy & Astrophysics

Perfectly parallel cosmological simulations using spatial comoving Lagrangian acceleration

F. Leclercq[1], B. Faure[2], G. Lavaux[3], B. D. Wandelt[3,4,5], A. H. Jaffe[1], A. F. Heavens[1], and W. J. Percival[6,7,8]

[1] Imperial Centre for Inference and Cosmology (ICIC) & Astrophysics Group, Imperial College London, Blackett Laboratory, Prince Consort Road, London SW7 2AZ, UK
e-mail: florent.leclercq@polytechnique.org
[2] AIM, CEA, CNRS, Université Paris-Saclay, Université Paris Diderot, Sorbonne Paris Cité, 91191 Gif-sur-Yvette, France
[3] CNRS, Sorbonne Université, UMR 7095, Institut d'Astrophysique de Paris, 98bis bd Arago, 75014 Paris, France
[4] Sorbonne Université, Institut Lagrange de Paris (ILP), 98 bis bd Arago, 75014 Paris, France
[5] Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, 10010 New York, NY, USA
[6] Waterloo Centre for Astrophysics, University of Waterloo, 200 University Ave W, Waterloo ON N2L 3G1, Canada
[7] Department of Physics and Astronomy, University of Waterloo, 200 University Ave W, Waterloo, ON N2L 3G1, Canada
[8] Perimeter Institute for Theoretical Physics, 31 Caroline St. North, Waterloo ON N2L 2Y5, Canada

Received 20 March 2020 / Accepted 15 May 2020

ABSTRACT

Context. Existing cosmological simulation methods lack a high degree of parallelism due to the long-range nature of the gravitational force, which limits the size of simulations that can be run at high resolution.
Aims. To solve this problem, we propose a new, perfectly parallel approach to simulate cosmic structure formation, which is based on the spatial COmoving Lagrangian Acceleration (sCOLA) framework.
Methods. Building upon a hybrid analytical and numerical description of particles' trajectories, our algorithm allows for an efficient tiling of a cosmological volume, where the dynamics within each tile is computed independently. As a consequence, the degree of parallelism is equal to the number of tiles. We optimised the accuracy of sCOLA through the use of a buffer region around tiles and of appropriate Dirichlet boundary conditions around sCOLA boxes.
Results. As a result, we show that cosmological simulations at the degree of accuracy required for the analysis of the next generation of surveys can be run in drastically reduced wall-clock times and with very low memory requirements.
Conclusions. The perfect scalability of our algorithm unlocks profoundly new possibilities for computing larger cosmological simulations at high resolution, taking advantage of a variety of hardware architectures.

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

1. Introduction

We live in the age of large astronomical surveys. These surveys detect and record tracers of cosmic structure across vast volumes of the Universe, using electromagnetic and gravitational waves. A non-exhaustive list includes optical and infrared imaging and spectroscopic surveys such as LSST (LSST Science Collaboration 2012), Euclid (Laureijs et al. 2011), DESI (DESI Collaboration 2016), and SPHEREx (SPHEREx Science Team 2018); catalogues and intensity maps from large radio surveys such as the square kilometer array (Square Kilometre Array Cosmology Science Working Group 2018) and its precursors; cluster catalogues from high-resolution observations of the microwave sky (Advanced ACTPol, Simon et al. 2018; SPTPol, Austermann et al. 2012; Simons Observatory, Simons Observatory Collaboration 2019, and CMB-S4); X-ray surveys such as the eROSITA mission (Merloni et al. 2012); as well as gravitational wave sirens across cosmological volumes with successive updates of (Advanced) LIGO (LIGO Scientific Collaboration 2015), Virgo (The Virgo Collaboration 2020) and LISA (Barausse et al. 2020). Whilst these data sets will be prodigious sources of scientific discovery across astrophysics, their enormous volume and dense sampling of cosmic structure will make them uniquely powerful when studying some of the deepest scientific mysteries of our time: the statistical properties of the primordial perturbations, the nature of dark matter, and the physical properties of dark energy. Indeed many of these surveys were conceived to address these questions.

Accomplishing this promise requires the ability to model these surveys in sufficient detail and with sufficient accuracy. All but the most simplistic models require the production of cosmological light-cone simulations. In particular, cosmological inferences often rely on large numbers of mock catalogues, which are used to construct unbiased estimators and study their statistical properties, such as covariance matrices. As surveys are getting deeper, these mock catalogues now need to represent a sizeable portion of the observable Universe, up to a redshift of $\sim$2–3 (e.g. $z = 2.3$ for the Euclid Flagship simulation[1]). Unfortunately, cosmological simulations put a heavy load on supercomputers. Even if only dark matter is included and resolution is minimised, they can require millions of CPU hours and hundreds of terabytes of disk space to solve the gravitational evolution of billions of particles and store the corresponding data. For instance, the DEUS-FUR simulation (Alimi et al. 2012),

[1] https://www.euclid-ec.org/?page_id=4133

containing 8192^3 particles in a box of 21 Gpc h^{-1} side length, required 10 million hours of CPU time and 300 TB of storage.

While computational needs are soaring, the performance of individual compute cores attained a plateau around 2015. Traditional hardware architectures are reaching their physical limit. Therefore, cosmological simulations cannot merely rely on processors becoming faster to reduce the computational time. Current hardware development focuses on increasing power efficiency[2] and solving problems of heat dissipation to allow packing a larger number of cores into each CPU. As a consequence, the performance gains of the world's top supercomputers are the result of a massive increase in the number of parallel cores, currently[3] to $O(10^5)$, and soon to $O(10^{6-7})$ in systems that are currently being built[4]. Hybrid architectures, where CPUs work alongside GPUs and/or reconfigurable chips such as FPGAs, add to the massive parallelism. In the exa-scale world, raw compute cycles are no longer the scarce resource. The challenge is to access the available computational power when Amdahl's law demonstrates that communication latencies kill the potential gains due to parallelisation (Amdahl 1967).

A way to embed high-resolution simulation of objects such as galaxy clusters, or even galaxies, in a cosmological context is through the use of varying particle mass resolution and the adaptive mesh refinement technique (AMR, Berger & Colella 1989). AMR is widely employed in grid-based simulation codes such as RAMSES (Teyssier 2002), ENZO (Bryan et al. 2014), FLASH (Fryxell et al. 2000), and AMIGA (Knebe & Doumler 2010). It is also used in MUSIC (Hahn & Abel 2011) to generate zoom-in initial conditions for simulations. The AMR technique, which uses multi-grid relaxation methods (e.g. Guillet & Teyssier 2011), allows focusing the effort on a specific region of the computational domain, but requires a two-way flow of information between small and large scales. More recently, leading computational cosmology groups have been developing sophisticated schemes to leverage parallel and hybrid computing architectures (Gonnet et al. 2013; Theuns et al. 2015; Aubert et al. 2015; Ocvirk et al. 2016; Potter et al. 2017; Yu et al. 2018; Garrison et al. 2019; Cheng et al. 2020).

Full simulations of large cosmological volumes, even limited to cold dark matter and at coarse resolution, involve multiple challenges. One of the main issues preventing their easy parallelisation is the long-range nature of gravitational interactions, which forestalls high-resolution, large-volume cosmological simulations. As a a response, much of the classical work in numerical cosmology focused on computational algorithms (tree codes, fast multipole methods, particle-mesh methods, and hybrids such as particle-particle–particle-mesh and tree–particle-mesh) that reduced the need for $O(N^2)$ all-to-all communications between N particles across the full computational volume.

While these algorithms are and remain the backbone of computational cosmology, they fail to fully exploit the physical scale hierarchy of cosmological perturbations. This hierarchy has first been used to push the results of N-body simulations to Universe scale for cosmic velocity fields (Strauss et al. 1995). At the largest scales, the dynamics of the Universe is not complicated, and in particular, is well-captured by Lagrangian Perturbation Theory (LPT; see Bouchet et al. 1995). Building upon this view, Tassev et al. (2015) introduced spatial COmoving Lagrangian Acceleration (sCOLA). This algorithm, using a hybrid analytical and numerical treatment of particles' trajectories, allows one to perform simulations without the need to substantially extend the simulated volume beyond the region of interest in order to capture far-field effects, such as density fluctuations due to super-box modes. The sCOLA proof-of-concept focused on one sub-box embedded into a larger simulation box.

In this paper, we extend the sCOLA algorithm and use it within a novel method for perfectly parallel cosmological simulations. To do so, we rely on a tiling of the full cosmological volume to be simulated, where each tile is evolved independently using sCOLA. The principal challenge for the accuracy of such simulations are the boundary conditions used throughout the evolution of tiles, which can introduce artefacts. In this respect, we introduce three crucial improvements with respect to Tassev et al. (2015): the use of a buffer region around each tile, the use of exact boundary conditions in the calculation of LPT displacements (which has the side benefit of reducing memory requirements), and the use of a Poisson solver with Dirichlet boundary conditions meant to approximate the exact gravitational potential around sCOLA boxes. The method proposed in this work shares similar goals with zoom-in simulation techniques, the main difference residing in the change of frame of reference introduced in sCOLA, which accounts for the dynamics of large scales without requiring flows of information during the evolution. On the other hand, our method is independent of the N-body integrator used to calculate the numerical part of particles' trajectories within each sCOLA box, and therefore, it cannot be related to specific approaches to do so, such as force-splitting. It is slightly approximate and more CPU-expensive than the corresponding "monolithic" simulation technique (chosen in this paper as tCOLA, Tassev et al. 2013), but has the essential advantage of perfect scalability. This scalability comes from the removal of any kind of communication among tiles after the initialisation of the simulation. As a consequence, for its major part, the degree of parallelism of the algorithm equals the number of tiles, which means that the workload is perfectly parallel (also called embarrassingly parallel). This property can be exploited to produce cosmological simulations in very short wall-clock times on a variety of hardware architectures, as we discuss in this paper.

After reviewing Lagrangian Perturbation Theory and its use within numerical simulations in Sect. 2, we describe our algorithm for perfectly parallel cosmological simulations in Sect. 3. In Sect. 4, we test the accuracy and speed of the algorithm with respect to reference simulations that do not use the tiling. We discuss the implications of our results for computational strategies to model cosmic structure formation, and conclude, in Sect. 5. Details regarding the implementation are provided in the appendices.

2. Cosmological simulations using Lagrangian perturbation theory

Throughout this section we denote by a the scale factor of the Universe. For simplicity, some of the equations are abridged. We reintroduce the omitted constants, temporal prefactors, and Hubble expansion in Appendix A.

Particle simulators are algorithms that compute the final position $\boldsymbol{x}$ and momentum $\boldsymbol{p} \equiv \mathrm{d}\boldsymbol{x}/\mathrm{d}a$ of a set of particles, given some initial conditions. They can also be seen as algorithms that compute a displacement field $\boldsymbol{\Psi}$, which maps the

[2] For example, Oak-Ridge National Laboratories' (ORNL) Summit machine has a typical power consumption of about 13 MW.
[3] https://www.olcf.ornl.gov/olcf-resources/compute-systems/summit/
[4] See for example ORNL's next supercomputer, Frontier: https://www.olcf.ornl.gov/wp-content/uploads/2019/05/frontier_specsheet.pdf

initial (Lagrangian) position $\boldsymbol{q}$ of each particle to its final (Eulerian) position $\boldsymbol{x}$, according to the classic equation (see e.g. Bernardeau et al. 2002, for a review)

$$\boldsymbol{x}(a) = \boldsymbol{q} + \boldsymbol{\Psi}(\boldsymbol{q}, a). \tag{1}$$

With this point of view, the outputs are $\boldsymbol{x}$ and $\boldsymbol{p} = \partial\boldsymbol{\Psi}/\partial a$.

2.1. Lagrangian perturbation theory (LPT)

In Lagrangian perturbation theory (LPT), the displacement field is given by an analytic equation which is used to move particles, without the need for a numerical solver. At second order in LPT, the displacement field is written

$$\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a) = \boldsymbol{\Psi}^{(1)}(\boldsymbol{q}, a) + \boldsymbol{\Psi}^{(2)}(\boldsymbol{q}, a), \tag{2}$$

where each of the terms is separable into a temporal and a spatial contribution deriving from a Lagrangian potential:

$$\boldsymbol{\Psi}^{(1)}(\boldsymbol{q}, a) = -D_1(a)\,\boldsymbol{\nabla}_{\boldsymbol{q}}\phi^{(1)}(\boldsymbol{q}), \tag{3}$$

$$\boldsymbol{\Psi}^{(2)}(\boldsymbol{q}, a) = D_2(a)\,\boldsymbol{\nabla}_{\boldsymbol{q}}\phi^{(2)}(\boldsymbol{q}). \tag{4}$$

In Eqs. (3) and (4), D_1 and D_2 are the growth factor and second-order growth factor, respectively. The Lagrangian potentials obey Poisson-like equations (Buchert et al. 1994):

$$\Delta_{\boldsymbol{q}}\phi^{(1)}(\boldsymbol{q}) = \delta_{\rm i}(\boldsymbol{q}), \tag{5}$$

$$\Delta_{\boldsymbol{q}}\phi^{(2)}(\boldsymbol{q}) = \sum_{i>j}\left[\phi^{(1)}_{,ii}\phi^{(1)}_{jj} - \left(\phi^{(1)}_{,ij}\right)^2\right], \tag{6}$$

where $\delta_{\rm i}(\boldsymbol{q})$ is the density contrast in the initial conditions, in Lagrangian coordinates, and the $\phi^{(1)}_{ij}$ are spatial second derivatives of $\phi^{(1)}$, i.e. $\phi^{(1)}_{ij} \equiv \partial^2\phi^{(1)}/\partial\boldsymbol{q}_i\partial\boldsymbol{q}_j$.

If only the first-order term is included in Eq. (2), the solution is known as the Zel'dovich approximation (Zel'dovich 1970).

2.2. Temporal comoving Lagrangian acceleration (tCOLA)

In contrast to the analytical equations of LPT, particle-mesh (PM) codes (see e.g. Klypin & Holtzman 1997) provide a fully numerical solution to the problem of large-scale structure formation. The equation of motion to be solved in a PM code reads schematically

$$\partial_a^2\boldsymbol{\Psi}(\boldsymbol{q}, a) = -\boldsymbol{\nabla}_{\boldsymbol{x}}\Phi(\boldsymbol{x}, a), \tag{7}$$

where the gravitational potential Φ satisfies the Poisson equation,

$$\Delta_{\boldsymbol{x}}\Phi(\boldsymbol{x}, a) = \delta(\boldsymbol{x}, a). \tag{8}$$

Here, $\delta(\boldsymbol{x}, a)$ is the density contrast at a scale factor a, which is obtained from the set of particles' positions $\{\boldsymbol{x}(a)\}$ through a density assignment operator that we denote B (typically a cloud-in-cell (CiC) scheme, see Hockney & Eastwood 1981):

$$\delta(\boldsymbol{x}, a) \equiv \mathrm{B}(\{\boldsymbol{x}(a)\}). \tag{9}$$

We denote by $\bar{\mathrm{B}}$ the corresponding interpolation operator, which is needed to obtain the accelerations of particles given the acceleration field on the grid:

$$\partial_a^2\boldsymbol{\Psi}(\{\boldsymbol{x}(a)\}) \equiv \bar{\mathrm{B}}(-\boldsymbol{\nabla}_{\boldsymbol{x}}\Phi). \tag{10}$$

The temporal COmoving Lagrangian Acceleration (tCOLA) algorithm seeks to decouple large and small scales by evolving large scales using analytic LPT results, and small scales using a numerical solver. This is achieved by splitting the Lagrangian displacement field into two contributions (Tassev & Zaldarriaga 2012):

$$\boldsymbol{\Psi}(\boldsymbol{q}, a) \equiv \boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a) + \boldsymbol{\Psi}_{\rm res}(\boldsymbol{q}, a), \tag{11}$$

where $\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a)$ is the LPT displacement field discussed in Sect. 2.1 and $\boldsymbol{\Psi}_{\rm res}(\boldsymbol{q}, a)$ is the residual displacement of each particle, as measured in a frame comoving with an "LPT observer", whose trajectory is given by $\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a)$. Using Eq. (11), it is possible to rewrite Eq. (7) as

$$\partial_a^2\boldsymbol{\Psi}_{\rm res}(\boldsymbol{q}, a) = -\boldsymbol{\nabla}_{\boldsymbol{x}}\Phi(\boldsymbol{x}, a) - \partial_a^2\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a). \tag{12}$$

The term $\partial_a^2\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a)$ can be thought of as a fictitious force acting on particles, caused by our use of a non-inertial frame of reference. Importantly, it can be computed analytically given the equations of Lagrangian perturbation theory.

The equations of motions (7) and (12) are usually integrated by the use of time-stepping techniques (see Appendix B). In the limit of zero time-steps used to discretise the left-hand side of Eq. (12), $\boldsymbol{\Psi}_{\rm res} = 0$ and tCOLA recovers the results of LPT; therefore, tCOLA always solves the large scales with an accuracy of at least that of LPT. In contrast, PM codes require many time-steps in Eq. (7) just to recover the value of the linear growth factor D_1. In the limit where the number of time-steps becomes large, tCOLA reduces to a standard PM code. In the intermediate regime (for $O(10)$ time-steps), tCOLA provides a good approximation to large-scale structure formation, at the expense of not solving the details of particle trajectories in deeply non-linear halos (see Tassev et al. 2013; Howlett et al. 2015; Leclercq et al. 2015; Koda et al. 2016; Izard et al. 2016, for further discussion). Since by construction, tCOLA always gets the large scales correct, contrary to a PM code, the trade-off between speed and accuracy only affects small scales.

2.3. Spatial comoving Lagrangian acceleration (sCOLA)

During large-scale structure formation, non-linearities appear at late times and/or at small scales. tCOLA (Eq. (12)) decouples LPT displacements and residual non-linear contributions "in time", so that, for a given accuracy, fewer time-steps are required to solve large-scale structure evolution than with a PM code. Following a similar spirit, the spatial COmoving Lagrangian Acceleration (sCOLA) framework decouples LPT displacements and residual non-linear contributions "in space", so that numerically evolved small scales can feel far-field effects captured analytically via LPT.

More specifically, for each particle in a volume of interest (the "sCOLA box") embedded in a larger cosmological volume (the "full box"), the equation of motion of particles, which reads for a traditional N-body problem

$$\partial_a^2\boldsymbol{\Psi}(\boldsymbol{q}, a) = \partial_a^2\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a) + \partial_a^2\boldsymbol{\Psi}_{\rm res}(\boldsymbol{q}, a) = \boldsymbol{F}(\boldsymbol{x}, a) \tag{13}$$

is replaced by

$$\partial_a^2\boldsymbol{\Psi}_{\rm res}(\boldsymbol{q}, a) = \boldsymbol{F}^{\rm sCOLA}(\boldsymbol{x}, a) - \partial_a^2\boldsymbol{\Psi}^{\rm sCOLA}_{\rm LPT}(\boldsymbol{q}, a). \tag{14}$$

$\partial_a^2\boldsymbol{\Psi}_{\rm res}(\boldsymbol{q}, a)$ is defined by Eq. (11) as the residual displacement with respect to the LPT observer of the full box, whose trajectory is given by $\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a)$. In Eq. (14), $\boldsymbol{\Psi}^{\rm sCOLA}_{\rm LPT}(\boldsymbol{q}, a)$ is the trajectory prescribed by solving LPT equations (see Sect. 2.1) in the sCOLA box. Note that $\boldsymbol{\Psi}^{\rm sCOLA}_{\rm LPT}(\boldsymbol{q}, a)$ may differ from $\boldsymbol{\Psi}_{\rm LPT}(\boldsymbol{q}, a)$,

depending on the assumptions made for the boundary conditions of the sCOLA box, discussed in Sect. 3.3. Denoting by $\mathcal{S} \subseteq [\![1, N]\!]$ the set of particles in the sCOLA box, the gravitational force, which in Eq. (13) reads

$$\boldsymbol{F}(\boldsymbol{x}_i, a) \equiv \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{\boldsymbol{x}_j(a) - \boldsymbol{x}_i(a)}{|\boldsymbol{x}_j(a) - \boldsymbol{x}_i(a)|^3}, \tag{15}$$

is replaced by

$$\boldsymbol{F}^{\mathrm{sCOLA}}(\boldsymbol{x}_i, a) \equiv \sum_{\substack{j\in\mathcal{S} \\ j\neq i}} \frac{\boldsymbol{x}_j(a) - \boldsymbol{x}_i(a)}{|\boldsymbol{x}_j(a) - \boldsymbol{x}_i(a)|^3}. \tag{16}$$

It is possible to evaluate $\boldsymbol{F}^{\mathrm{sCOLA}}(\boldsymbol{x}, a)$, and thus to solve Eq. (14), like Eq. (13), using any numerical gravity solver, such as particle-particle–particle-mesh, tree codes, or AMR. In this paper, we choose to focus on evaluating forces via a PM scheme. In this case, the equation of motion of particles in sCOLA reads schematically (Tassev et al. 2015)

$$\partial_a^2 \boldsymbol{\Psi}_{\mathrm{res}}(\boldsymbol{q}, a) = -\boldsymbol{\nabla}_{\boldsymbol{x}}^{\mathrm{sCOLA}} \Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a) - \partial_a^2 \boldsymbol{\Psi}_{\mathrm{LPT}}^{\mathrm{sCOLA}}(\boldsymbol{q}, a). \tag{17}$$

The gravitational potential in the sCOLA box, $\Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a)$, obeys the near-field version of the Poisson equation,

$$\Delta_{\boldsymbol{x}}^{\mathrm{sCOLA}} \Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a) = \delta^{\mathrm{sCOLA}}(\boldsymbol{x}, a). \tag{18}$$

The superscript "sCOLA" over the gradient and Laplacian operators, $\boldsymbol{\nabla}_{\boldsymbol{x}}^{\mathrm{sCOLA}}$ and $\Delta_{\boldsymbol{x}}^{\mathrm{sCOLA}}$, mean that they are restricted to the sCOLA box (contrary to that of Eqs. (8) and (12)). Over the density contrast $\delta^{\mathrm{sCOLA}}(\boldsymbol{x}, a)$, the superscript means that only particles in the sCOLA box $\{\boldsymbol{x}(a)\}_{\mathrm{sCOLA}} \equiv \{\boldsymbol{x}_i(a)\}_{i\in\mathcal{S}}$ (instead of the full box) are used within the density assignment $\mathrm{B}^{\mathrm{sCOLA}}$, i.e.

$$\delta^{\mathrm{sCOLA}}(\boldsymbol{x}, a) \equiv \mathrm{B}^{\mathrm{sCOLA}}\left(\{\boldsymbol{x}(a)\}_{\mathrm{sCOLA}}\right). \tag{19}$$

Contrary to tCOLA, which is an exact rewriting of the equations of motion of a PM code, sCOLA potentially involves approximations for the calculation of each quantity and operator with a superscript "sCOLA" instead of its full box equivalent. As a proof of concept, Tassev et al. (2015) showed that under certain circumstances, sCOLA provides a good approximation for the evolution of one sCOLA box embedded into a larger full box. As discussed in the introduction, we aim at generalising this result by using sCOLA within multiple sub-volumes of a full simulation box.

3. Algorithm for perfectly parallel simulations using sCOLA

In this section, we describe an algorithm for cosmological simulations using sCOLA, for which the time evolution of independent Lagrangian sub-volumes is perfectly parallel, without any communication. A functional block diagram representing the main steps and their dependencies is given in Fig. 1. An illustration of the different grids appearing in the algorithm is presented in Fig. 2, and Table 1 provides the nomenclature of some of the different variables appearing in this section.

We work in a cubic full box of side length L with periodic boundary conditions, populated by N_{p}^3 particles initially at the nodes $\{\boldsymbol{q}\}$ of a regular Lagrangian lattice. We seek to compute the set of final positions $\{\boldsymbol{x}(a_{\mathrm{f}})\}$ and momenta $\{\boldsymbol{p}(a_{\mathrm{f}})\}$ at final scale factor a_{f}. The model equations are reviewed in Appendix A. The time-stepping of these equations consists of a series of "kick" and "drift" operations and is discussed in Appendix B.

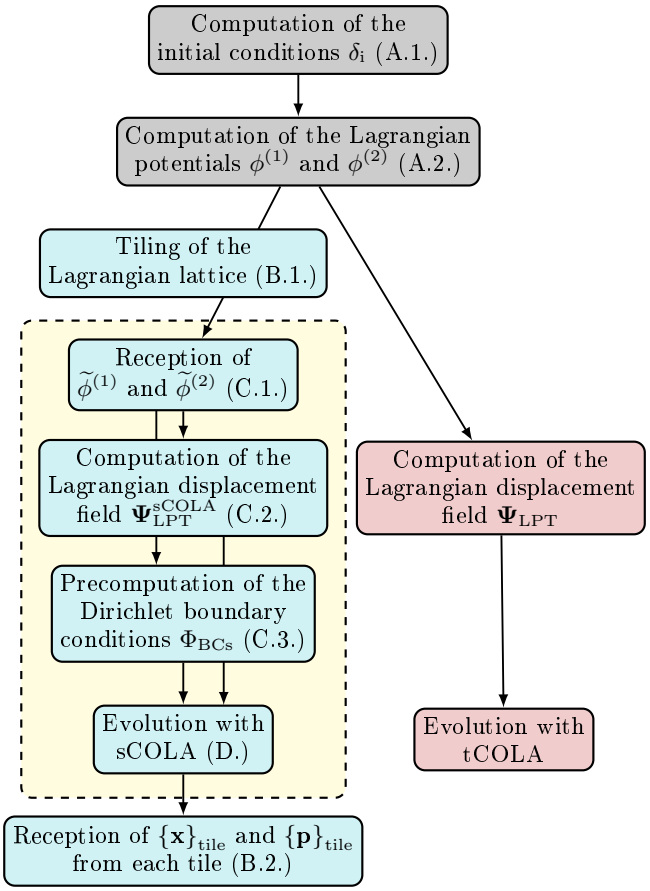

Fig. 1. Functional diagram of sCOLA (*left*) versus tCOLA (*right*). The grey boxes are common steps. sCOLA specific steps are represented in blue, and tCOLA specific steps in red. The yellow rectangle constitutes the perfectly parallel section, within which no communication is required with the master process or between processes. Arrows represent dependencies, and references to the main text are given between parentheses.

We approximate the Laplacians $\Delta_{\boldsymbol{x}}$, $\Delta_{\boldsymbol{q}}$ and gradient operators $\boldsymbol{\nabla}_{\boldsymbol{x}}$, $\boldsymbol{\nabla}_{\boldsymbol{q}}$ by finite difference approximation (FDA) at order 2, 4, or 6. The coefficients of the finite difference stencils in configuration and in Fourier space are given for example in table 1 in Hahn & Abel (2011). We note $N_{\mathrm{ghost}} = 1, 2, 3$ if FDA is taken at order 2, 4, 6, respectively.

3.1. Initial conditions and Lagrangian potentials

Before the perfectly parallel section, two initialisation steps are performed by the master process in the full box.

A.1. The first step is to generate the initial density contrast δ_{i} in the full box, on a cubic grid of N^3 cells (the "LPT grid", represented in red in the left panel of Fig. 2). This step can be done via the standard convolution approach (e.g. Hockney & Eastwood 1981), given the specified initial power spectrum.

A.2. The second step is to compute the Lagrangian potentials $\phi^{(1)}(\boldsymbol{q})$ and $\phi^{(2)}(\boldsymbol{q})$ on the LPT grid in the full box, which is achieved by solving Eqs. (5) and (6).

If initial phases are generated in Fourier space, the Zel'dovich approximation (i.e. the calculation of $\phi^{(1)}$) requires only one inverse fast Fourier transform (FFT) on the LPT grid.

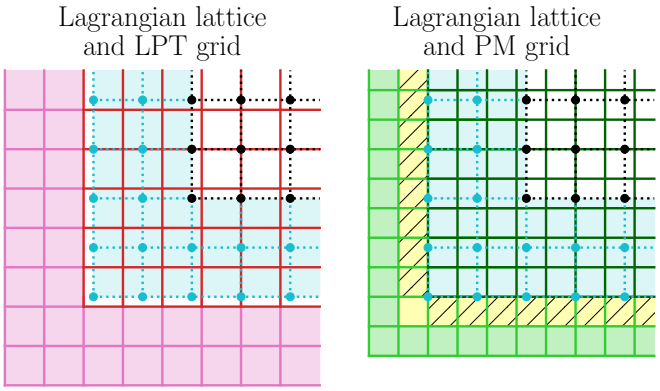

Fig. 2. Illustration of the different grids used within sCOLA. The Lagrangian lattice is represented by dashed lines. For each tile, central particles (in black) are surrounded by buffer particles (in cyan), which are ignored at the end of the evolution. The corresponding buffer region in other grids is represented in cyan. The *left panel* represents the "LPT grid" on which Lagrangian potentials $\widetilde{\phi}^{(1)}$ and $\widetilde{\phi}^{(2)}$ are defined. The central region has $N^3_{\rm sCOLA}$ grid points (in red) and is padded by $2N_{\rm ghost}$ cells in each direction (pink region). The *right panel* shows the "PM grid" on which the density contrast $\delta^{\rm sCOLA}$, the gravitational potential $\Phi^{\rm sCOLA}$, and the accelerations $-\boldsymbol{\nabla}^{\rm sCOLA}_{\boldsymbol{x}}\Phi^{\rm sCOLA}$ are defined. The density contrast is defined only in the central region (which has $N^3_{\rm g}$ grid points, in dark green). The gravitational potential is padded by $2N_{\rm ghost}$ cells in each direction (light green and yellow regions), and the gridded accelerations only by $N_{\rm ghost}$ cells in each direction (yellow region). Solving the Poisson equation requires Dirichlet boundary conditions in six layers of $N_{\rm ghost}$ cells, denoted as hatched regions. For simplicity of representation, we have used here $N_{\rm ghost} = 1$.

Table 1. Nomenclature of symbols used in the present article.

Symbol	Meaning
N	LPT grid size in the full box
$N_{\rm p}$	Lagrangian lattice size in the full box
$N_{\rm tiles}$	Number of tiles in each direction
$N_{\rm p,tile}$	Number of particles per direction in each tile
$L_{\rm tile}$	Physical size of each tile
$N_{\rm p,buffer}$	Number of buffer particles per direction
$L_{\rm buffer}$	Physical size of the buffer region
$N_{\rm p,sCOLA}$	Number of particles per direction in each sCOLA box
$L_{\rm sCOLA}$	Physical size of each sCOLA box
$N_{\rm tile}$	LPT grid portion covering each tile
$N_{\rm sCOLA}$	LPT grid portion covering each sCOLA box
$N_{\rm ghost}$	Number of ghost cells depending on FDA
$N_{\rm g}$	PM grid size in each sCOLA box
r	Over-simulation factor
p	Parallelisation potential factor

For the second-order potential, the source term on the right-hand side of Eq. (6) has to be computed from $\phi^{(1)}$; this can either be done in Fourier space (for a cost of six inverse FFTs) or in configuration space via finite differencing (for a cost of nine one-dimensional gradient operations). In both cases, the calculation of $\phi^{(2)}$ from its source then requires one forward and one inverse FFT.

These few FFTs in the full box are the most hardware-demanding requirement of the algorithm (particularly in terms of memory), and the only step which is not distributed and suitable for grid computing. These FFTs may however be performed on a cluster of computers with fast interconnection suitable for Message Passing Interface (Frigo & Johnson 2005; Johnson et al. 2008).

3.2. Tiling and buffer region

B.1. After having computed the Lagrangian potentials, the master process splits the Lagrangian lattice (of size $N^3_{\rm p}$) into $N^3_{\rm tiles}$ cubic tiles (we require that $N_{\rm p}$ is a multiple of $N_{\rm tiles}$). Tiles are constructed to be evolved independently; therefore the main, perfectly parallel region of the algorithm starts here.

To minimise artefacts due to boundary effects (see Sect. 3.4), each tile is surrounded by a "buffer region" in Lagrangian space. This buffer region consists of $N_{\rm p,buffer}$ particles in each direction, so that each sCOLA box contains a total of $N^3_{\rm p,sCOLA}$ particles, where $N_{\rm p,sCOLA} \equiv N_{\rm p,tile} + 2N_{\rm p,buffer}$ and $N_{\rm p,tile} \equiv N_{\rm p}/N_{\rm tiles}$. Corresponding physical sizes are $L_{\rm tile} \equiv L\,N_{\rm p,tile}/N_{\rm p}$, $L_{\rm buffer} \equiv L\,N_{\rm p,buffer}/N_{\rm p}$, and $L_{\rm sCOLA} \equiv L\,N_{\rm p,sCOLA}/N_{\rm p}$. The fraction of the full Lagrangian lattice assigned to one child sCOLA process is represented by dotted lines in Fig. 2. Particles of the tile are represented in black, and particles of the buffer region are represented in cyan.

The sCOLA box is chosen to encompass the tile and its buffer region. We define the over-simulation factor r as the ratio between the total volume simulated in all sCOLA boxes and the target simulation volume, i.e.

$$\begin{aligned} r \quad &\equiv \quad \frac{N^3_{\rm tiles}N^3_{\rm p,sCOLA}}{N^3_{\rm p}} = \frac{N^3_{\rm tiles}(N_{\rm p,tile}+2N_{\rm p,buffer})^3}{N^3_{\rm p}} \\ &= \quad \frac{N^3_{\rm tiles}L^3_{\rm sCOLA}}{L^3} = \frac{N^3_{\rm tiles}(L_{\rm tile}+2L_{\rm buffer})^3}{L^3}. \end{aligned} \tag{20}$$

Since all sCOLA boxes can be evolved independently, the degree of parallelism of the algorithm is equal to the number of sCOLA boxes, $N^3_{\rm tiles}$. We call the "parallelisation potential factor" the quantity $p \equiv N^3_{\rm tiles}/r$, which balances the degree of parallelism with the amount of over-simulation. It is also

$$p = \frac{N^3_{\rm p}}{N^3_{\rm p,sCOLA}} = \frac{L^3}{L^3_{\rm sCOLA}}. \tag{21}$$

For each sCOLA box, the corresponding child process computes the set of final positions $\{\boldsymbol{x}\}_{\rm sCOLA}$ and momenta $\{\boldsymbol{p}\}_{\rm sCOLA}$.

B.2. At the end of the evolution, each child process sends the set of final positions $\{\boldsymbol{x}\}_{\rm tile}$ and momenta $\{\boldsymbol{p}\}_{\rm tile}$ of particles of the tile back to the master process. Particles of the buffer region are ignored. The master process then "untiles" the simulation by gathering the results from all the tiles.

3.3. Initial operations in the sCOLA boxes

A few steps are required in each sCOLA box before starting the evolution per se.

C.1. The sCOLA box receives the relevant portion of $\phi^{(1)}(\boldsymbol{q})$ and $\phi^{(2)}(\boldsymbol{q})$ from the master process. This is the only communication required with the master process before sending back the results at the end of the evolution.

The portion of the LPT grid received by each process from the master process corresponds to the full spatial region covered by the sCOLA box, plus an additional padding of $2N_{\rm ghost}$ cells in each direction. We denote by $\widetilde{\phi}^{(1)}(\boldsymbol{q})$ and $\widetilde{\phi}^{(2)}(\boldsymbol{q})$ the parts of $\phi^{(1)}(\boldsymbol{q})$ and $\phi^{(2)}(\boldsymbol{q})$ received from the master process (we avoid the superscript "sCOLA" since no approximation is involved at

this stage). They are defined on a grid of size $(N_{\rm sCOLA}+4N_{\rm ghost})^3$, where

$$N_{\rm tile} \equiv \left\lceil N_{\rm p,tile}\frac{N}{N_{\rm p}}\right\rceil,\ N_{\rm sCOLA} \equiv N_{\rm tile} + 2\left\lceil N_{\rm p,buffer}\frac{N}{N_{\rm p}}\right\rceil \tag{22}$$

($\lceil\cdot\rceil$ denotes the ceiling function). An illustration is provided in Fig. 2, left panel. There, the portion of the LPT grid corresponding to the sCOLA box, of size $N_{\rm sCOLA}$ in each direction, is represented in red and the padding region, of size $2N_{\rm ghost}$ in each direction, is represented in pink.

C.2. The sCOLA process locally computes the required time-independent LPT vectors $\boldsymbol{\Psi}_1^{\rm sCOLA}$ and $\boldsymbol{\Psi}_2^{\rm sCOLA}$ via finite differencing in configuration space and interpolation to particles' positions.

The ghost cells included around $\widetilde{\phi}^{(1)}(\boldsymbol{q})$ and $\widetilde{\phi}^{(2)}(\boldsymbol{q})$ in the sCOLA box ensure that the proper boundary conditions are used when applying the gradient operator $\boldsymbol{\nabla}_{\boldsymbol{q}}^{\rm sCOLA}$ in configuration space to get the LPT displacements on the grid. This step "consumes" $N_{\rm ghost}$ layers of ghost cells in each direction, so that the grid of LPT displacements has a size of $(N_{\rm sCOLA}+2N_{\rm ghost})^3$. To use again the proper boundary conditions when going from the LPT grid to particles' positions, another $N_{\rm ghost}$ layers of ghost cells is consumed by the interpolation operator $\bar{\rm B}^{\rm sCOLA}$. The use of the exact boundary conditions at each of these two steps ensures that $\boldsymbol{\nabla}_{\boldsymbol{q}}^{\rm sCOLA} = \boldsymbol{\nabla}_{\boldsymbol{q}}$ and $\bar{\rm B}^{\rm sCOLA} = \bar{\rm B}$. Therefore, by construction, $\boldsymbol{\Psi}_1^{\rm sCOLA} \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}^{\rm sCOLA}\widetilde{\phi}^{(1)}(\boldsymbol{q})$ and $\boldsymbol{\Psi}_2^{\rm sCOLA} \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}^{\rm sCOLA}\widetilde{\phi}^{(2)}(\boldsymbol{q})$ in the sCOLA box are always the same as $\boldsymbol{\Psi}_1 \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}\phi^{(1)}(\boldsymbol{q})$ and $\boldsymbol{\Psi}_2 \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}\phi^{(2)}(\boldsymbol{q})$ in the full box (as would be computed by the master process). Consequently, we do not keep track of both $\boldsymbol{\Psi}_{1,2}^{\rm sCOLA}$ and $\boldsymbol{\Psi}_{1,2}$, contrary to Tassev et al. (2015). In addition to being simpler, this scheme has the practical advantage of saving six floating-point numbers per particle in memory (three in the case of the Zel'dovich approximation).

C.3. The sCOLA process precomputes the Dirichlet boundary conditions $\Phi_{\rm BCs}$ that will be used at each calculation of the gravitational potential during the sCOLA evolution.

For each sCOLA box, we define a particle-mesh grid of size $N_{\rm g}^3$ (the "PM grid", represented in dark green in the right panel of Fig. 2). The PM grid defines the force resolution; it should be equal to or finer than the LPT grid ($N_{\rm g} \geq N_{\rm sCOLA}$). Before starting the evolution with sCOLA, each process precomputes the Dirichlet boundary conditions that will be required by the Poisson solver at each value of the scale factor $a_{\rm K}$. This calculation takes as input the initial gravitational potential $\widetilde{\phi}^{(1)}(\boldsymbol{q})$ and outputs $\Phi_{\rm BCs}(\boldsymbol{x}, a_{\rm K})$ for each $a_{\rm K}$, defined on the PM grid with a padding of $2N_{\rm ghost}$ cells around the sCOLA box in each direction (light green and yellow regions in Fig. 2, right panel). The approximation involved in this step is further discussed in Sect. 3.4.2.

3.4. Evolution of sCOLA boxes

Each sCOLA box is then evolved according to the scheme reviewed in Sect. 2.3 and Appendices A and B. Two specific approximations are needed to compute the operators and quantities with a superscript "sCOLA"; we now discuss the choices that we made.

3.4.1. Density assignment ($\mathrm{B}^{\rm sCOLA}$)

As mentioned in Sect. 2.3, only particles of the sCOLA box should contribute to $\delta^{\rm sCOLA}(\boldsymbol{x}, a)$. For particles that are fully in the sCOLA box, density assignment can be chosen as the same operation as would be used in a PM or tCOLA code (typically, a CiC scheme). A question is what to do with particles that have (partially) left the sCOLA box during the evolution, while keeping the requirement of no communication between boxes: this constitutes the only difference between the operators B and $\mathrm{B}^{\rm sCOLA}$. Possible choices include artificially periodising the sCOLA box (which is clearly erroneous) or stopping particles at its boundaries (which does not conserve momentum). Both of these choices assign the entire mass carried by the set of sCOLA particles $\mathcal{S}$ to the PM grid, but result in artefacts in the final conditions, if the buffer region is not large enough.

An alternative choice is simply to limit the (Eulerian) PM grid volume where we compute $\delta^{\rm sCOLA}(\boldsymbol{x}, a)$ to the (Lagrangian) sCOLA box, including central and buffer regions. In practice, this means ignoring the fractional particle masses that the CiC assignment would have deposited to grid points outside the sCOLA box. We have found in our tests that this choice gives the smallest artefacts of the three choices considered[5]. We note that (partially) erasing some particles' mass is an approximation that is only used in the $\mathrm{B}^{\rm sCOLA}$ operator to evaluate the source term in the Poisson equation, and therefore only affects the force calculation. The number of particles, both within each sCOLA process ($N^3_{\rm p,sCOLA}$) and in the full simulation ($N^3_{\rm p}$), is left unchanged during the evolution. Therefore, mass is always conserved both within each sCOLA process and within the full volume.

3.4.2. Gravitational potential ($\Delta_{\boldsymbol{x}}^{\rm sCOLA}$, $\boldsymbol{\nabla}_{\boldsymbol{x}}^{\rm sCOLA}$ and $\bar{\rm B}^{\rm sCOLA}$)

Poisson solver ($\Delta_{\boldsymbol{x}}^{\rm sCOLA}$). To make sure that differences between $\Phi^{\rm sCOLA}(\boldsymbol{x}, a)$ and $\Phi(\boldsymbol{x}, a)$ are as small as possible, we make use of a Poisson solver with Dirichlet boundary conditions, instead of assuming periodic boundary conditions. Such a Poisson solver uses discrete sine transforms (DSTs) instead of FFTs, and requires the boundary values of Φ in six planes (west, east, south, north, bottom, top) surrounding the PM grid (see Appendix C). These planes have a thickness of $N_{\rm ghost}$ cells (depending on the value of the FDA used to approximate the Laplacian); they are represented by hatched regions in Fig. 2, right panel. At each scale factor $a_{\rm K}$ when the computation of accelerations is needed, the Dirichlet boundary conditions are extracted from the precomputed $\Phi_{\rm BCs}(\boldsymbol{x}, a_{\rm K})$ (step C.3., see Sect. 3.3).

Ideally, $\Phi_{\rm BCs}(\boldsymbol{x}, a_{\rm K})$ should be the exact, non-linear gravitational potential in the full volume at $a_{\rm K}$, $\Phi(\boldsymbol{x}, a_{\rm K})$. However, knowing this quantity would require having previously run the monolithic simulation in the full volume, which we seek to avoid. In this paper, we rely instead on the linearly-evolving potential (LEP) approximation (Brainerd et al. 1993; Bagla & Padmanabhan 1994), namely

$$\Phi_{\rm BCs}(\boldsymbol{x}, a_{\rm K}) \approx \Phi_{\rm LEP}(\boldsymbol{x}, a_{\rm K}) \equiv D_1(a_{\rm K})\,\widetilde{\phi}^{(1)}(\boldsymbol{x}). \tag{23}$$

The idea behind this approximation is that the gravitational potential is dominated by long-wavelength modes, and therefore it ought to obey linear perturbation theory to a better approximation than the density field.

In Eq. (23), we have assumed that the linear growth factor D_1 is normalised to unity at the scale factor corresponding to the initial conditions. The precomputation of $\Phi_{\rm BCs}$ in step C.3. is therefore an interpolation from the LPT grid to the PM grid and a simple scaling with $D_1(a_{\rm K})$.

[5] There is a certain symmetry to this choice, since particles that would have moved into the buffer region from the outside are also neglected in the force calculation, due to the lack of communication between different sCOLA boxes.

Table 2. Different setups used to test the accuracy and speed of our sCOLA algorithm.

L [Mpc h^{-1}]	N_p	N	N_tiles	$N_\mathrm{p,tile}$	L_tile [Mpc h^{-1}]	$N_\mathrm{p,buffer}$	L_buffer [Mpc h^{-1}]	N_g	r	p
200	512	256	16	32	12.5	32	12.5	97	27	151.70
			8	64	25	32	12.5	129	8	64
			8	64	25	64	25	193	27	18.96
			4	128	50	32	12.5	193	3.38	18.96
			4	128	50	64	25	257	8	8
			4	128	50	128	50	385	27	2.37
			2	256	100	32	12.5	321	1.95	4.10
			2	256	100	64	25	385	3.38	2.37
1000	1024	512	16	64	62.5	14	13.7	93	2.97	1378.91
			16	64	62.5	26	25.4	117	5.95	687.90
			16	64	62.5	40	39.1	145	11.39	359.59
			16	64	62.5	64	62.5	193	27	151.70
			8	128	125	10	9.8	149	1.55	331.22
			8	128	125	20	19.5	169	2.26	226.45
			8	128	125	30	29.3	189	3.17	161.59
			8	128	125	50	48.8	229	5.65	90.59

The output of the Poisson solver is the gravitational potential $\Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a_\mathrm{K})$ on the PM grid, in the interior of the sCOLA box (dark green grid points in Fig. 2, right panel). Consistently with the treatment above, $\Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a_\mathrm{K})$ is padded using the values of $\Phi_\mathrm{BCs}(\boldsymbol{x}, a_\mathrm{K})$ in $2N_\mathrm{ghost}$ cells around the PM grid, in each direction (light green and yellow regions in Fig. 2, right panel).

Therefore, the only difference between $\Delta_{\boldsymbol{x}}^{\mathrm{sCOLA}}$ and $\Delta_{\boldsymbol{x}}$ resides in using the LEP instead of the true, non-linear gravitational potential at the boundaries of the sCOLA box.

Accelerations ($\boldsymbol{\nabla}_{\boldsymbol{x}}^{\mathrm{sCOLA}}$ *and* $\bar{\mathrm{B}}^{\mathrm{sCOLA}}$). Given the gravitational potential $\Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a_\mathrm{K})$, accelerations are computed by finite differencing in configuration space and interpolation to particles' positions, similarly to step C.2. (see Sect. 3.3). The application of $\boldsymbol{\nabla}_{\boldsymbol{x}}^{\mathrm{sCOLA}}$ consumes N_ghost cells, so that accelerations are obtained on the PM grid with a padding of N_ghost cells (yellow region in Fig. 2, right panel). Interpolation from the grid to particles' position (the $\bar{\mathrm{B}}^{\mathrm{sCOLA}}$ operator) further consumes N_ghost cells.

As for the Laplacian, the only difference between $\boldsymbol{\nabla}_{\boldsymbol{x}}^{\mathrm{sCOLA}}$ and $\boldsymbol{\nabla}_{\boldsymbol{x}}$, and $\bar{\mathrm{B}}^{\mathrm{sCOLA}}$ and $\bar{\mathrm{B}}$, resides in using the LEP in $\Phi^{\mathrm{sCOLA}}(\boldsymbol{x}, a_\mathrm{K})$ instead of the true, non-linear gravitational potential at the boundaries of the sCOLA box.

4. Accuracy and speed

We implemented the perfectly parallel sCOLA algorithm described in Sect. 3 in the SIMBELMYNË code (Leclercq et al. 2015), publicly available[6] (see also Leclercq 2015, appendix B, for technical details on the implementation of the PM and tCOLA models in SIMBELMYNË). This section describes some tests of the accuracy and speed of the new sCOLA algorithm. Since our implementation, relying on evaluating forces with a PM scheme, introduces some additional approximations with respect to tCOLA, we compare our results to that of corresponding monolithic tCOLA simulations. The accuracy of tCOLA with respect to more accurate gravity solvers has been characterised in the earlier literature (Tassev et al. 2013; Howlett et al. 2015; Leclercq et al. 2015; Koda et al. 2016; Izard et al. 2016). The question of comparing the accuracy of our sCOLA algorithm to full N-body simulations would require building in a full N-body integrator for the sCOLA boxes (see Eqs. (14) and (16)); this subject is left for future research.

Throughout the paper, we adopt the ΛCDM model with *Planck* 2015 cosmological parameters: $h = 0.6774$, $\Omega_\Lambda = 0.6911$, $\Omega_\mathrm{b} = 0.0486$, $\Omega_\mathrm{m} = 0.3089$, $n_\mathrm{S} = 0.9667$, $\sigma_8 = 0.8159$ (Planck Collaboration XIII 2016, page 31, Table 4, last column). The initial power spectrum is computed using the Eisenstein & Hu (1998, 1999) fitting function.

We base our first tests on a periodic box of comoving side length $L = 200\,\mathrm{Mpc}\,h^{-1}$ populated with $N_\mathrm{p}^3 = 512^3$ dark matter particles. For all operators, we use FDA at order 2. The LPT grid has $N^3 = 256^3$ voxels. Particles are evolved to redshift $z = 19$ using 2LPT. For all runs, we use 10 time-steps linearly-spaced in the scale factor to evolve particles from $z = 19$ ($a_\mathrm{i} = 0.05$) to $z = 0$ ($a_\mathrm{f} = 1$) (see Appendix B)[7]. For tCOLA, the PM grid, covering the full box, has 512^3 voxels. For sCOLA, we use eight different setups, with various parameters $\{N_\mathrm{tiles}, N_\mathrm{p,tile}, L_\mathrm{tile}, N_\mathrm{p,buffer}, L_\mathrm{buffer}, N_\mathrm{g}, r, p\}$ given in the first part of Table 2.

To assess more extensively the impact of using sCOLA on large scales, we used a second ensemble of simulations with the following differences: a box with comoving side length of $L = 1\,\mathrm{Gpc}\,h^{-1}$, $N_\mathrm{p} = 1024^3$ particles, a LPT grid with $N^3 = 512^3$ voxels, and a PM grid of 1024^3 voxels for tCOLA. For sCOLA, we use eight different setups given in the second part of Table 2.

4.1. Qualitative assessments

The redshift-zero density field is estimated by assigning all particles to the LPT grid using the CiC scheme. Results for the $200\,\mathrm{Mpc}\,h^{-1}$ box are shown in Fig. 3. There, the bottom right panel shows the reference tCOLA density field and other panels show the differences between sCOLA and tCOLA results, for the eight different setups. Some qualitative observations can be made: when artefacts are visible in the sCOLA results, they mainly affect over-dense regions of the cosmic web (filaments and halos), whereas under-dense regions are generally better recovered. Artefacts are of two types: the position of a structure

[6] https://bitbucket.org/florent-leclercq/simbelmyne/

[7] This means that in the case of our new sCOLA algorithm, we use COLA both "in space and time" (see Tassev et al. 2015).

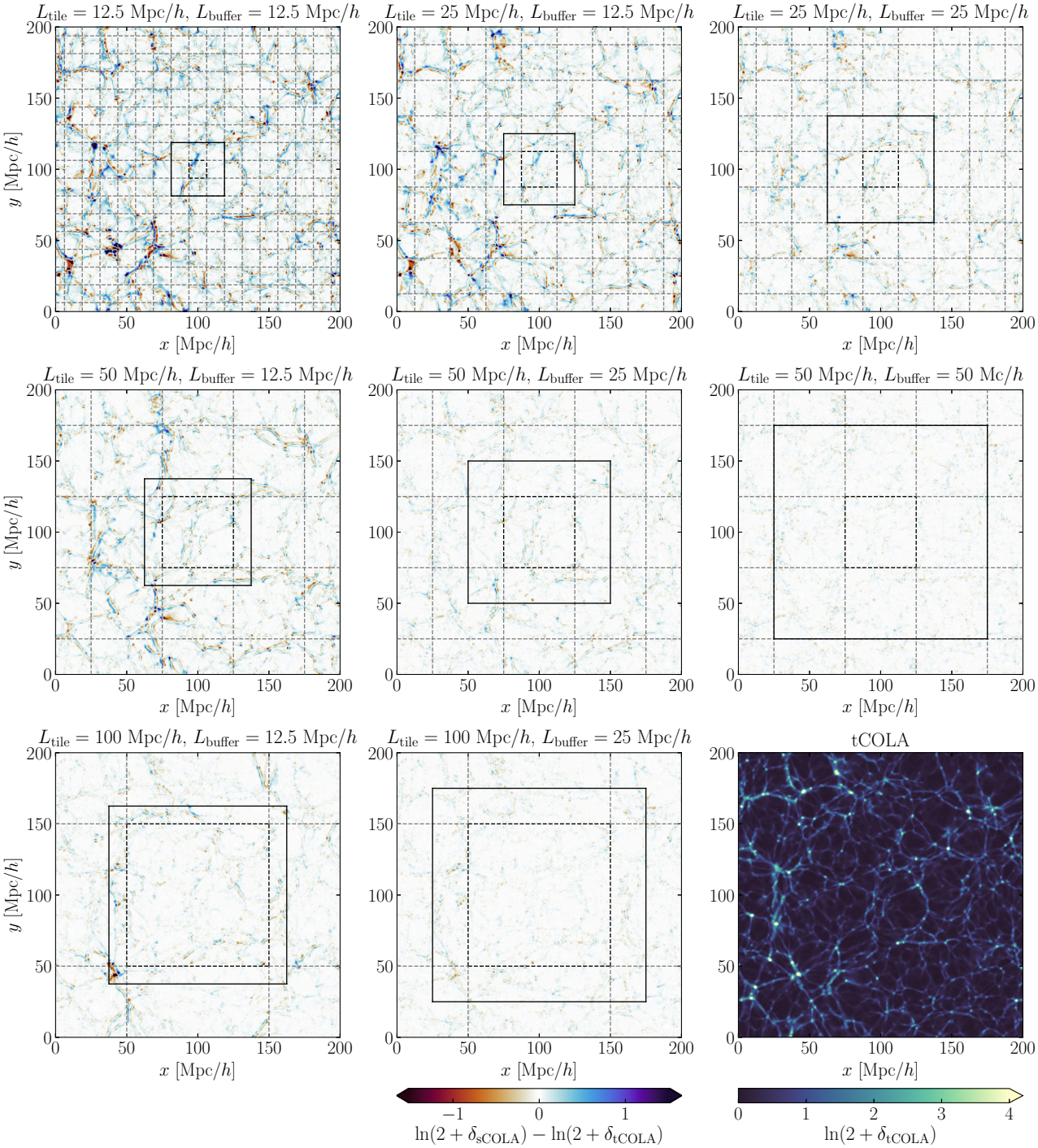

Fig. 3. Qualitative assessment of the redshift-zero density field from sCOLA for different tilings and buffer sizes, with respect to tCOLA. The *bottom right panel* shows the reference tCOLA density field in a 200 Mpc h^{-1} box with periodic boundary conditions (the quantity represented is $\ln(2 + \delta_{\rm tCOLA})$ where $\delta_{\rm tCOLA}$ is the density contrast). Other panels show the difference between sCOLA and tCOLA density fields, $\ln(2 + \delta_{\rm sCOLA}) - \ln(2 + \delta_{\rm tCOLA})$, for different sizes of tile and buffer region, as indicated above the panels. The tiling is represented by dashed lines, and the central tile's buffer region is represented by solid lines. In the third dimension, the slices represented intersect the central tile at its centre. As can be observed in this figure, artefacts are predominantly located close to the boundaries of tiles; they are reduced with increasing tile size and buffer region size.

(usually a filament) can be imprecise due to a misestimation of bulk motions (this is visible as a "dipole" in Fig. 3); or the density (usually of halos) can be over- or under-estimated (this is visible as a "monopole" in Fig. 3). In all setups, artefacts are predominantly located close to the boundaries of tiles (represented as dashed lines) and are less visible in the centre of tiles. This can be easily understood given that the approximations made all concern the behaviour at the boundaries of sCOLA boxes. At fixed size for the buffer region, the correspondence between sCOLA and tCOLA density fields improves with increasing tile size. A minimum tile size of about 50 Mpc h^{-1} seems necessary to limit the misestimation of halo densities ("monopoles" in Fig. 3). At low redshift, this scale is in the mildly non-linear regime, where LPT starts to break down; therefore, the LPT frame is

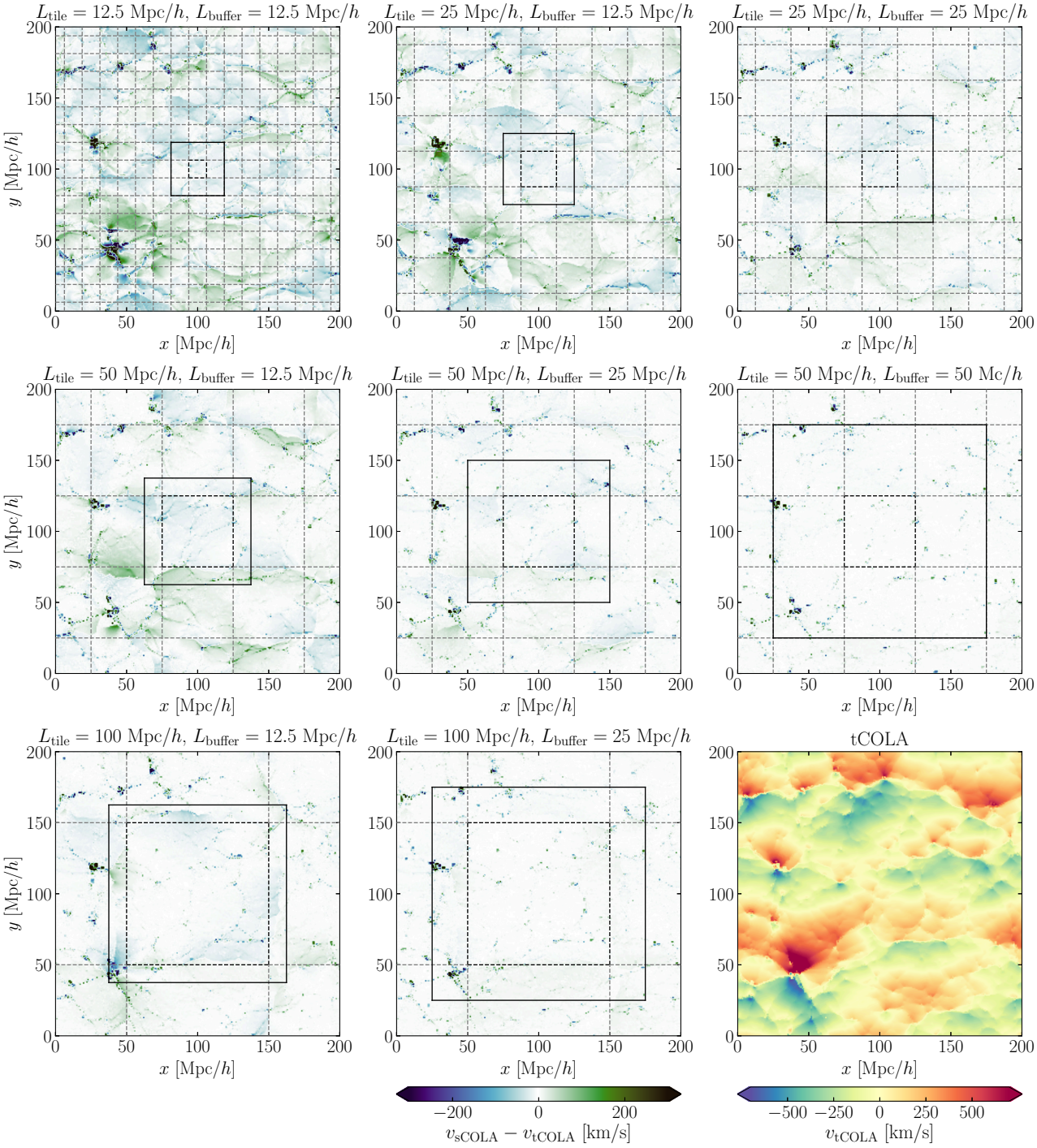

Fig. 4. Same as Fig. 3, but for one component of the velocity field, in km/s. Bulk flows are correctly captured if tiles and their buffer regions are large enough. Residual differences inside halos can be observed, but they are expected due to the limited number of time-steps, rendering both tCOLA and sCOLA velocities inaccurate in the deeply non-linear regime.

inaccurate for particles, and the requirement of no communication between tiles leads to mispredicted clustering. As expected, at fixed tile size, the results are improved by increasing the buffer region around tiles: in each sCOLA box, boundary approximations are pushed farther away from the central region of interest. A good compromise between reducing artefacts and increasing the size of buffer regions seems to be found for a buffer region of 25 Mpc h^{-1}, which corresponds roughly to the maximum distance travelled by a particle from its initial to its final position. In particular, the setup $L_{\rm tile} = 50\,{\rm Mpc}\,h^{-1}$, $L_{\rm buffer} = 25\,{\rm Mpc}\,h^{-1}$ leads to a satisfactory approximation of the tCOLA density with a parallelisation potential factor $p = 8$.

In a similar fashion, the velocity field is estimated on the LPT grid from particle information, using the simplex-in-cell estimator (Hahn et al. 2015; Leclercq et al. 2017). Using phase-space information, this estimator accurately captures the velocity field, even in regions sparsely sampled by simulation particles. Results for the 200 Mpc h^{-1} box are shown in Fig. 4, where one component of the tCOLA velocity field $v_{\rm tCOLA}$ (in km s^{-1}) is shown in the bottom right panel. Other panels show the velocity error in

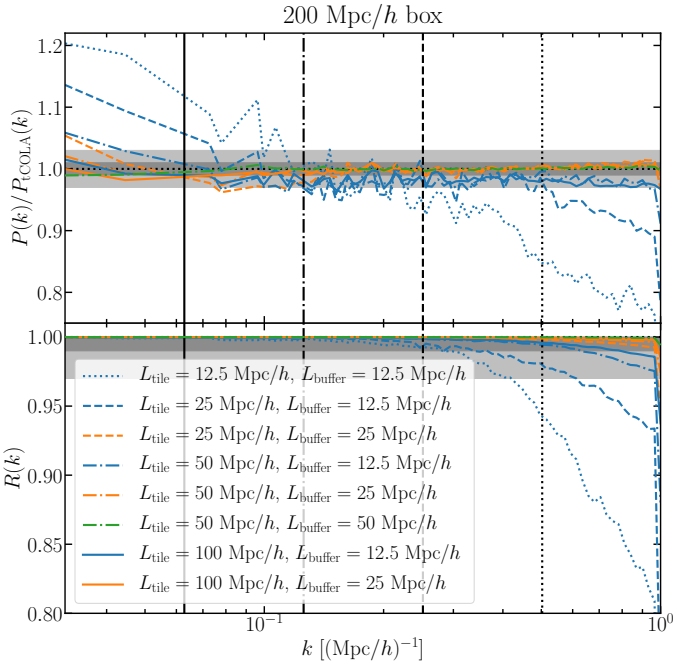

Fig. 5. Power spectrum relative to tCOLA (*top panel*) and cross-correlation with respect to tCOLA (*bottom panel*) of redshift-zero sCOLA density fields, in a 200 Mpc h^{-1} box containing 512^3 dark matter particles. Different sizes for the tiles (represented by different line styles) and buffer regions (represented by different colours) are used, as indicated in the legend. The vertical lines show the respective fundamental mode of different tiles, the light grey bands correspond to 3% accuracy, and the dark grey bands to 1% accuracy.

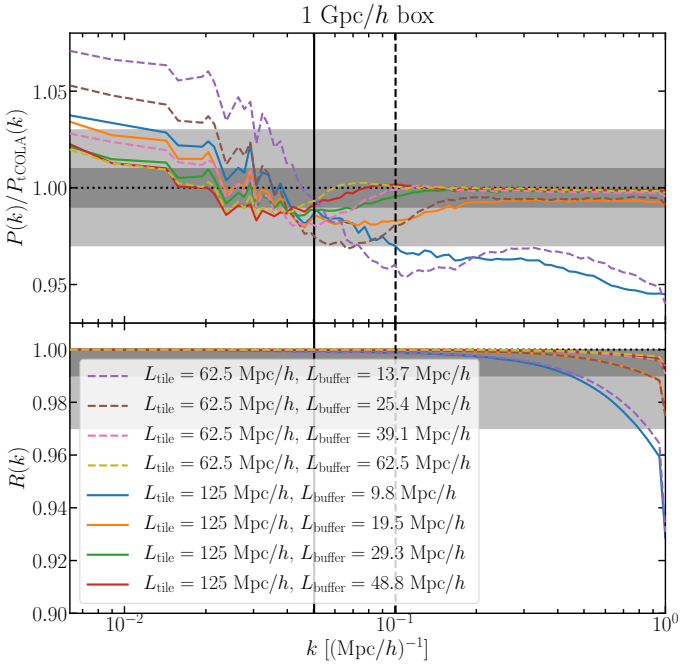

Fig. 6. Same as Fig. 5, but in a 1 Gpc h^{-1} box containing 1024^3 particles.

sCOLA, $v_{\mathrm{sCOLA}} - v_{\mathrm{tCOLA}}$ in km s^{-1}. Differences between tCOLA and sCOLA velocity fields are of two kinds: misestimation of bulk flows (visible as light, spatially extended regions in Fig. 4), or misestimation of particle velocities inside halos (visible as dark spots in Fig. 4). We do not interpret the second kind of differences as errors made by our sCOLA algorithm: indeed, motions within virialised regions are not captured accurately by any simulation using only ten time-steps, even by tCOLA in the full box. Therefore, only the first kind of differences, that is, the misestimation of coherent bulk motions is physically interpretable. In this respect, the same behaviour as for density fields can be observed: artefacts are mostly located at the boundaries of tiles, and they are reduced with increasing tile size and buffer region size, with safe minima of $L_{\mathrm{tile}} \gtrsim 50\,\mathrm{Mpc}\,h^{-1}$ and $L_{\mathrm{buffer}} \gtrsim 25\,\mathrm{Mpc}\,h^{-1}$, respectively.

4.2. Summary statistics

In this section, we turn to a more quantitative assessment of our results, by checking the power spectrum of final density fields and their cross-correlation to the tCOLA density field. Even if final density fields are non-Gaussian, two-point statistics (auto- and cross-spectra) are expected to be sensitive to the approximations made in our sCOLA algorithm, which involves both local and non-local operations in configuration space.

According to Huterer & Takada (2005) or Audren et al. (2013), in the best cases, observational errors for a Euclid-like survey are typically of order 3% for $k < 10^{-2}\,(\mathrm{Mpc}\,h^{-1})^{-1}$. These results do not account for any of the systematic uncertainties linked to selection effects or contamination of the clustering signal by foregrounds. At smaller scales, theoretical uncertainties take over, reaching 1% and above for $k > 10^{-1}\,(\mathrm{Mpc}\,h^{-1})^{-1}$. In addition, the impact of baryonic physics is still largely uncertain, some models predicting an impact of at least 10% at $k = 1\,(\mathrm{Mpc}\,h^{-1})^{-1}$ (e.g. van Daalen et al. 2011; Chisari et al. 2018; Schneider et al. 2019). Any data model involving our sCOLA algorithm will be subject to these uncertainties. For this reason, we aim for no better than 3% to 1% accuracy at all scales up to $k = 1\,(\mathrm{Mpc}\,h^{-1})^{-1}$, for any two-point measurement of clustering.

More precisely, we work with $P(k)$ and $R(k)$, defined for two density contrast fields δ and $\delta' = \delta_{\mathrm{tCOLA}}$, with our Fourier transform convention, by

$$\delta_{\mathrm{D}}(\boldsymbol{k}-\boldsymbol{k}')P(k) \equiv (2\pi)^{-3}L^6\,\langle\delta^*(\boldsymbol{k})\delta(\boldsymbol{k}')\rangle\,, \tag{24}$$

$$\delta_{\mathrm{D}}(\boldsymbol{k}-\boldsymbol{k}')R(k) \equiv \frac{\langle\delta^*(\boldsymbol{k})\delta'(\boldsymbol{k}')\rangle}{\sqrt{\langle\delta^*(\boldsymbol{k})\delta(\boldsymbol{k}')\rangle\,\langle\delta'^*(\boldsymbol{k})\delta'(\boldsymbol{k}')\rangle}}, \tag{25}$$

where δ_{D} is a Dirac delta distribution. For the estimation of $P(k)$ and $R(k)$, we use 100 logarithmically-spaced k-bins from the fundamental mode of the box $k_{\mathrm{min}} \equiv 2\pi/L$ to $k = 1\,(\mathrm{Mpc}\,h^{-1})^{-1}$.

In Figs. 5 and 6, we plot the power spectrum of sCOLA density fields divided by the power spectrum of the reference tCOLA density field, $P_{\mathrm{sCOLA}}(k)/P_{\mathrm{tCOLA}}(k)$ (upper panels) and the cross-correlation between sCOLA and tCOLA density fields, $R(k)$ (bottom panels), for our 200 Mpc h^{-1} (Fig. 5) and 1 Gpc h^{-1} box (Fig. 6). The grey horizontal bands represent the target accuracies of 3% and 1%, and the vertical lines mark the fundamental modes of the tiles, $k_{\mathrm{tile}} \equiv 2\pi/L_{\mathrm{tile}}$, for the different values of L_{tile} used.

Figure 5 quantitatively confirms the considerations of Sect. 4.1. Both the amplitudes (as probed by $P(k)/P_{\mathrm{tCOLA}}(k)$) and the phase accuracy (as probed by $R(k)$) of sCOLA simulations are improved with increasing tile size, for a fixed buffer region (different line styles, same colours). For a fixed tile size, results are also improved by increasing the size of the buffer region (same line styles, different colours). Remarkably, all setups yield perfect phase accuracy at large scales ($R(k) = 1$ for $k \leq 0.2\,(\mathrm{Mpc}\,h^{-1})^{-1}$), even when the amplitude of corresponding modes deviates from the tCOLA result. Defects at

small scales (lack of power and inaccurate phases) are only observed for the smallest tile sizes and are fixed by increasing the size of buffer region. This effect can be interpreted in Lagrangian coordinates: when the Lagrangian volume forming a halo is divided among different tiles that do not exchange particles, and if the buffer region is too small to contain the rest of the halo, the resulting structure is then split and underclustered in Eulerian coordinates. In this respect, preferring a sCOLA box size ($L_{\rm sCOLA} \equiv L_{\rm tile}+2L_{\rm buffer}$) of at least 100 Mpc h^{-1} (and therefore $L_{\rm tile} \gtrsim 50\,{\rm Mpc}\,h^{-1}$, $L_{\rm buffer} \gtrsim 25\,{\rm Mpc}\,h^{-1}$, in most situations) seems to be sensible. A more difficult issue is the amplitude of large-scale modes, for $k < k_{\rm tile}$. These are sensitive to the tiling if buffer regions around tiles are too small. A safe requirement also seems to be $L_{\rm buffer} \gtrsim 25\,{\rm Mpc}\,h^{-1}$. Putting everything together, in our 200 Mpc h^{-1} box, three setups reach 3% accuracy in amplitude and phases at all scales: $\{L_{\rm tile} = 50\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 25\,{\rm Mpc}\,h^{-1}\}$ (discussed already in Sect. 4.1); $\{L_{\rm tile} = 100\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 25\,{\rm Mpc}\,h^{-1}\}$; and $\{L_{\rm tile} = 50\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 50\,{\rm Mpc}\,h^{-1}\}$. The last-mentioned performs even better, reaching 1% accuracy at all scales, but at the price of over-simulating the volume by a larger factor.

Figure 6 shows the same diagnostics for a 1 Gpc h^{-1} box, where the qualitative behaviour is the same as before. It confirms the requirement $L_{\rm buffer} \gtrsim 25\,{\rm Mpc}\,h^{-1}$ to get sufficient accuracy at high k. The question of the accuracy reached at the largest scales is then jointly sensitive to $L_{\rm tile}$ and L. In our tests, the setups $\{L_{\rm tile} = 62.5\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 39.1\,{\rm Mpc}\,h^{-1}\}$ and $\{L_{\rm tile} = 125\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 29.3\,{\rm Mpc}\,h^{-1}\}$ yield 3% accurate results at all scales, and the setups $\{L_{\rm tile} = 62.5\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 62.5\,{\rm Mpc}\,h^{-1}\}$ and $\{L_{\rm tile} = 125\,{\rm Mpc}\,h^{-1}, L_{\rm buffer} = 48.8\,{\rm Mpc}\,h^{-1}\}$ almost reach 1%-level precision at all scales. We note that the two different boxes have different mass resolutions, which confirms that requirements for tile and buffer region sizes should be expressed in physical size.

4.3. Tests of the approximations

As discussed in Sect. 3.4, two approximations are introduced in our sCOLA algorithm with respect to a monolithic tCOLA approach. These concern density assignment in the interior of sCOLA boxes (approximation D.1.) and the gravitational potential at the boundaries of sCOLA boxes (approximation D.2.). In this section, we test the impact of these approximations on final results, using two-point statistics as diagnostic tools. For this test we use our sCOLA run with $L = 200\,{\rm Mpc}\,h^{-1}$, $N_{\rm p} = 512^3$, 64 tiles ($N_{\rm tiles} = 4$, $N_{\rm p,tile} = 128$) and $N_{\rm p,buffer} = 32$ (i.e. $L_{\rm tile} = 50\,{\rm Mpc}\,h^{-1}$, $L_{\rm buffer} = 12.5\,{\rm Mpc}\,h^{-1}$). We choose a small buffer size on purpose, to be sensitive to the approximations made.

Let us denote by $\delta_{\rm int}$ the density contrast in the interior of sCOLA boxes and by $\Phi_{\rm BCs}$ the gravitational potential at the boundaries of sCOLA boxes. As discussed in Sect. 3.4, our algorithm involves an approximation regarding particles leaving the sCOLA box during the evolution, yielding $\delta^{\rm sCOLA}$, and relies on the LEP approximation at the boundaries. It therefore uses

$$\delta_{\rm int} = \delta^{\rm sCOLA} \quad \text{and} \quad \Phi_{\rm BCs} = \Phi_{\rm LEP}. \tag{26}$$

Everything else being fixed, we ran three investigative sCOLA simulations using respectively,

$$\delta_{\rm int} = \delta \quad \text{and} \quad \Phi_{\rm BCs} = \Phi_{\rm LEP}, \tag{27}$$

$$\delta_{\rm int} = \delta^{\rm sCOLA} \quad \text{and} \quad \Phi_{\rm BCs} = \Phi, \tag{28}$$

$$\delta_{\rm int} = \delta \quad \text{and} \quad \Phi_{\rm BCs} = \Phi, \tag{29}$$

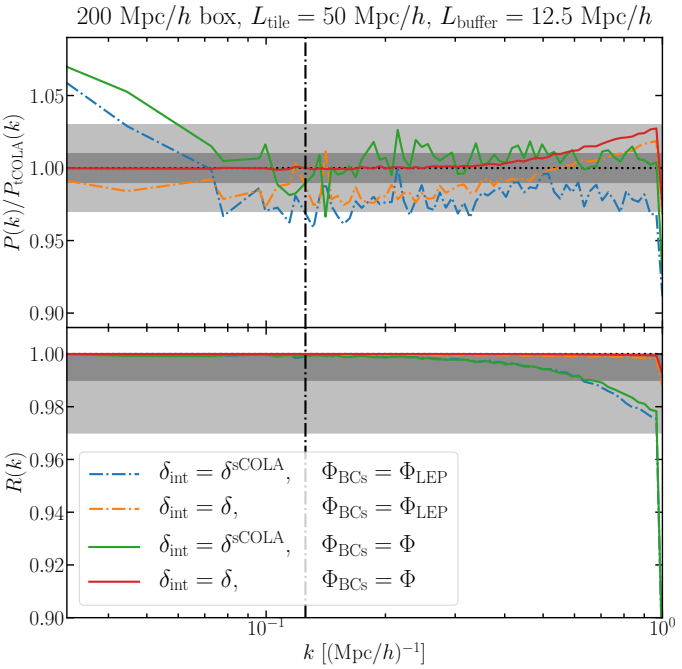

Fig. 7. Tests of the approximations made in sCOLA for the density field and the gravitational potential. As in Fig. 5, the diagnostic tools are the power spectrum relative to tCOLA (*top panel*) and the cross-correlation with tCOLA (*bottom panel*). Our sCOLA algorithm uses the approximate interior density field $\delta^{\rm sCOLA}$ and the LEP approximation for the boundary gravitational potential (dash-dotted blue line). In other simulations, as indicated in the legend, we use the true density field δ and/or the true gravitational potential Φ at the boundaries. The approximation made for the density field dominates, especially at large scales.

where δ is the "true" density contrast and Φ is the "true" gravitational potential, extracted at each time-step from the corresponding tCOLA simulation.

Figure 7 shows the auto- and cross-spectra of resulting sCOLA density fields, with respect to the reference tCOLA result. The use of $\delta_{\rm int} = \delta$ yields by construction $R(k) = 1$ at all scales, as can be checked from the bottom panel. The setup given by Eq. (29) is rid of the two approximations; it is therefore a consistency check: one should retrieve the tCOLA result if no bias is introduced by the tiling and different Poisson solver. As expected, Fig. 7 shows that our implementation recovers the tCOLA result at all scales, with only a small excess of power at $k > 0.4\,({\rm Mpc}\,h^{-1})^{-1}$ explained by the slightly higher force resolution of the sCOLA run with respect to tCOLA (the PM grid cell sizes are 0.3886 and 0.3906 Mpc h^{-1}, respectively).

The setups given by Eqs. (27) and (28) allow disentangling the impact of approximations D.1. and D.2. In the standard run (Eq. (26)), averaging over tiles and timesteps, ∼0.43% of the 512^3 particles, all of which belonging to the buffer region, do not deposit all of their mass in the calculation of $\delta^{\rm sCOLA}$, but ∼76.5% on average. This number only slightly increases with time (from ∼0.35% at $a = 0.05$ to ∼0.47% at $a = 1$); in other simulations, we have found that it has a stronger dependence on the mass resolution and on the surface of sCOLA boxes. Regarding the accuracy of the LEP approximation, the ratio of the power spectra of $\Phi - \Phi_{\rm LEP}$ and of Φ goes to zero at early times and large scales, and stays below 12% for all scales with wavenumber $k \leq 2\pi/L_{\rm sCOLA}$ at $a = 1$. As can be observed in Fig. 7, although using the non-linear gravitational potential instead of the LEP improves both $P(k)$ and $R(k)$ for the final density field

at all scales with wavenumber $k > 7 \times 10^{-2}\,(\mathrm{Mpc}\,h^{-1})^{-1}$, it does not remove the $\gtrsim$5% bias in amplitude at the largest scales. On the contrary, using the true density contrast solves this problem and yields a 3% accurate result at all scales, which is remarkable given the small buffer size used in this case (the over-simulation factor is only $r = 3.38$).

We conclude from these tests that the approximation made regarding the density field (D.1.) has more impact than the one regarding the gravitational potential (D.2.), especially on the largest modes. This result is consistent with the standard paradigm for structure formation, where the density contrast undergoes severe non-linearity at small scales and late times, while the gravitational potential evolves very little. It also suggests that future improvements of our algorithm should focus on finding a better approximation for δ^{sCOLA}, rather than Φ_{BCs}.

4.4. Computational cost

One of the main motivations for our perfectly parallel algorithm based on sCOLA is to be able to run very large volume simulations at reasonably high resolution. A detailed analysis of the speed and computational cost of our algorithm, as implemented in Simbelmynë, is therefore beyond the intent of this paper. However, in this section we discuss some performance considerations based on a sCOLA run with $L = 1\,\mathrm{Gpc}\,h^{-1}$, $N_{\mathrm{p}} = 1024^3$, 512 tiles ($N_{\mathrm{tiles}} = 8$, $N_{\mathrm{p,tile}} = 128$), $N_{\mathrm{p,buffer}} = 30$ (i.e. $L_{\mathrm{tile}} = 125\,\mathrm{Mpc}\,h^{-1}$, $L_{\mathrm{buffer}} = 29.3\,\mathrm{Mpc}\,h^{-1}$), $N_{\mathrm{g}} = 199$; and the corresponding monolithic tCOLA simulation. In this case, the over-simulation factor is $r \approx 3.17$ and the parallelisation potential factor is $p \approx 161.59$. To compare the theoretical parallelisation potential factor and the realised parallelisation efficiency, we use one process for tCOLA and 512 processes for sCOLA. Each process is run on a node with 32 cores using OpenMP parallelisation.

One of the main advantages of our sCOLA algorithm lies in its reduced memory consumption. In Fig. 8 (first row), we show the memory requirements for the calculation of LPT potentials in the full box (common for tCOLA and sCOLA), for the evolution of the full box with tCOLA, and for the evolution of each sCOLA box, all in single-precision floating-point format. LPT requires eight grids of size N^3 (one for the initial conditions, one for the Zel'dovich potential, and six for the second-order term), occupying ~4.3 GB. Evolution with tCOLA requires one integer and 12 floating-point numbers per particle (their identifier, their position $\boldsymbol{x}$, their momentum $\boldsymbol{p}$, and the vectors $\boldsymbol{\Psi}_1$ and $\boldsymbol{\Psi}_2$), plus a PM grid of 1024^3 voxels, for a total of ~60.1 GB. Within each box, sCOLA requires the same memory per particle (but with $N^3_{\mathrm{p,sCOLA}} \ll N^3_{\mathrm{p}}$), a PM grid of size N^3_{g}, and some overhead for Dirichlet boundary conditions. The total is around 400 MB per sCOLA box with the setup considered here.

In the second row of Fig. 8, we show the overall cost of tCOLA versus sCOLA, both in terms of CPU time (middle left panel) and wall-clock time (middle right panel). The key feature of our algorithm is that, although the overall CPU time needed is unavoidably higher than with tCOLA, the wall-clock time spent can be drastically reduced. This owes to the degree of parallelism of our algorithm, which is equal to the number of sCOLA boxes. In particular, if as many processes as sCOLA boxes can be allocated (512 in this case), the overall wall-clock time is determined by the initial full box operations (common with tCOLA, see Sect. 3.1), plus the cost of evolving only *one* sCOLA box (an average of 30.9 wall-clock seconds on 32 cores in this test). This is what is shown in the middle right panel of Fig. 8. The

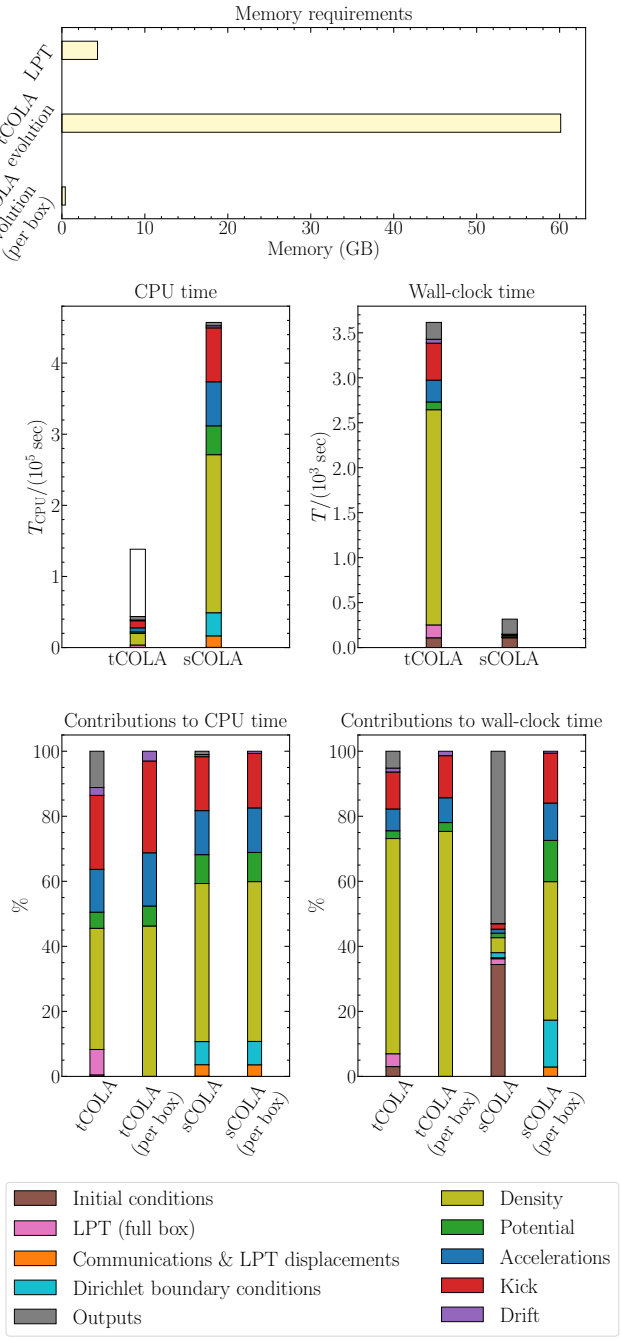

Fig. 8. Memory requirements (*first row*) and timings for two corresponding tCOLA and sCOLA simulations. Although the CPU time required is higher for sCOLA, the memory consumption and wall-clock time are significantly reduced with respect to tCOLA, due to the perfectly parallel nature of most computations (*second row*). In the *middle left panel*, the height of the white bar shows the hypothetical cost of running tCOLA for the same volume as simulated with sCOLA, when taking buffer regions into account. The relative contributions of different operations, as detailed in the legend, is shown in the *third row*. The main difference in computational cost in sCOLA with respect to tCOLA comes from the use of DSTs instead of FFTs, which makes the evaluation of the potential significantly more expensive.

wall-clock time reduction factor is ≈93 for the evolution only (≈11 when accounting for initialisation and writing outputs). Compared to the parallelisation potential factor $p \approx 162$, this number means that sCOLA-specific operations and the larger

fractional parallelisation overhead in sCOLA boxes do not significantly hamper the perfectly parallel nature of the code.

The increased CPU time needed with sCOLA (see Fig. 8, middle left panel) is partly due to the necessity of over-simulating the volume of interest by a factor $r > 1$ for accuracy. For comparison with the sCOLA CPU time, the height of the white bar shows the tCOLA CPU time multiplied by r. The rest of the difference in CPU time principally comes form the fact that simulations with our variant of sCOLA are intrinsically more expensive than with tCOLA for a periodic volume of the same size. This point is further discussed below.

In the third row of Fig. 8, we show the various relative contributions to CPU time and wall-clock time, both for full tCOLA/sCOLA runs and per tCOLA/sCOLA box. The generations of the initial conditions (brown, step A.1.) and writing of outputs to disk (grey) are common to tCOLA and sCOLA and have an overall fixed cost. LPT calculations in the full box (pink) consist of computing the Lagrangian potentials and the particle-based LPT displacements in tCOLA, but are limited to computing the Lagrangian potentials in the full box in the case of sCOLA (step A.2.). These full-box operations are only showed in the bars labelled "tCOLA" and "sCOLA". Within each box, the different operations are evaluating the density field (yellow), solving the Poisson equation to get the gravitational potential (green), differentiating the gravitational potential to get the accelerations (blue), "kicking" particles (red), and "drifting" particles (purple). sCOLA further requires some specific operations within each box: communicating with the master process (steps B.1., B.2., and C.1.), calculating the particle-based LPT displacements (step C.2.), grouped in Fig. 8 and shown in orange; and pre-computing the Dirichlet boundary conditions with the LEP approximation (step C.3., cyan). sCOLA-specific operations do not contribute more than 10% of the CPU and wall-clock times per box.

A notable difference between evolving a given box with sCOLA or with tCOLA resides in the higher cost of evaluating the potential (green): in this case, 9% of CPU time and 13% of wall-clock time with sCOLA versus 6% of CPU time and 3% of wall-clock time with tCOLA. This effect is due to the use of DSTs, required by the Poisson solver with Dirichlet boundary conditions (see Sect. 3.4 and Appendix C), instead of FFTs. Indeed, depending on the size of the PM grid, the evaluation of DSTs can be the computational bottleneck of our algorithm (up to 60% of overall CPU time is some of our runs), as opposed to the evaluation of the density field (e.g. via CiC) in traditional tCOLA or PM codes (37% of overall CPU time). For this reason, within each setup, we recommend performing experiments to find a PM grid size giving a good compromise between force accuracy and computational efficiency. In particular, it is strongly preferable that $N_\mathrm{g}+1$ not contain large prime factors (this number appears in the basis functions of sine transforms, see Appendix C.2). Throughout this paper, we ensured that $N_\mathrm{g}+1$ is always even, while keeping roughly the same force resolution as the corresponding tCOLA simulation. We note that our choice of $N_\mathrm{g}+1 = 200$ in the present test, combined with the use of a power of two for the PM grid in the monolithic tCOLA run, favours tCOLA in the comparison of CPU times. The sCOLA CPU time shown in the middle left panel of Fig. 8 could be further optimised by making $N_\mathrm{g}+1$ a power of two in sCOLA boxes.

5. Discussion and conclusion

5.1. Discussion

The principal computational challenge of the gravitational N-body problem is the long-range nature of the gravitational force. Our sCOLA approach enables perfectly parallel computations and therefore opens up profoundly new possibilities for how to compute large-scale cosmological simulations. We discuss these, some consequences and possible future directions in the following.

Gravity and physics models. It is important to note that the sCOLA algorithm introduced in this work is general, and not limited to the gravity model used here: while we focused on a tCOLA particle-mesh implementation to evolve the sCOLA tiles, this choice was designed to facilitate the assessment of tiling artefacts against monolithic tCOLA runs. Nonetheless, any N-body method, such as particle-particle–particle-mesh, tree methods or AMR, could be used to evolve each tile. In particular, since the sCOLA approach separates quasi-linear and non-linear scales, there is no need to cut off the computation on small scales. In concert with the approaches discussed below, this fact can be exploited to perform very high-resolution, fully non-linear simulations in cosmological volumes. In this case, the spatial decoupling due to sCOLA would render computations possible that would otherwise be prohibitive.

Similar comments apply to including non-gravitational physics: since hydrodynamical or other non-gravitational forces are typically much more local than gravitational interactions, there are no algorithmic barriers to including them in each sCOLA tile[8].

Construction of light-cones and mock catalogues. The decoupling of computational volumes achieved by our approach means that each sCOLA box can be run completely independently. Therefore, it is not necessary to define a common final redshift for all tiles. This means that to compute a cosmological light-cone, only a single tile (the one containing the observer) needs to be run to redshift zero. Since the volume on the light-cone increases rapidly with redshift, the vast majority of tiles would only have to be run until they intersect the light-cone at high redshift. In monolithic N-body simulations, most of the computational time is spent at low redshift, since the local time-step of simulations decreases with the local dynamical time. Our approach would therefore greatly accelerate the time needed to complete light-cone simulations, by scheduling tiles in order of the redshift to which they should run (and therefore in reverse order of expected computational time), aiding load-balancing.

The construction of light-cones for surveys with large aspect ratios, such as pencil-beam surveys, can further benefit from sCOLA. Indeed, tiles that do not intersect the three-dimensional survey window do not need to be run at all for the construction of mock catalogues. In such a case, the algorithm will still capture the effects of large-scale transverse modes, even if the simulated volume is not substantially increased with respect to the survey volume.

Low memory requirements. sCOLA divides the computational volume into much smaller tiles and vastly reduces the memory footprint of each independent sCOLA tile computation, as shown in Sect. 4.4. As an example, simulating a $(16\,\mathrm{Gpc}\,h^{-1})^3$ volume containing 8192^3 particles to achieve a mass resolution of $10^{12.5}\,M_\odot$ requires ∼19.8 TB of RAM with a PM code and ∼33.0 TB of RAM with tCOLA. The setup $\{L_\mathrm{tile} = 62.5\,\mathrm{Mpc}\,h^{-1}, L_\mathrm{buffer} = 62.5\,\mathrm{Mpc}\,h^{-1}\}$ would break down the problem into 256^3 tiles, each with $(3 \times 32)^3$ particles and a

[8] A potential exception is long-range radiative transport of energetic (X-ray or gamma ray) photons, requiring a non-trivial extension of the approach.

memory footprint of ∼53 MB. This has important consequences, which we explore in the following.

The very modest memory requirement of our algorithm opens up multiple possibilities to accelerate the computation: even on traditional systems, the entire computation of each sCOLA tile would fit entirely into the L3 cache of a multi-core processor. This would cut out the slowest parts of the memory hierarchy, leading to a large potential performance boost and reducing code complexity. Even more promising, many such tiles could be evolved entirely independently on GPU accelerators, or even dedicated FPGAs, taking advantage of hybrid architectures of modern computational platforms while reducing the need to develop sophisticated code to manage task parallelism. At this scale, each tile computation would even fit comfortably on ubiquitous small computational platforms such as mobile phones.

Grid computing. The perfect scalability achieved by our approach means that large N-body simulations can even be run on very inexpensive, strongly asynchronous networks designed for large throughput computing. An extreme example would be participatory computing platforms such as Cosmology@Home[9], where tens of thousands of users donate computational resources. The use of such platforms would be particularly suited to light-cone computations, as described above. Even if running the low-redshift part necessitates dedicated hardware, other workers could efficiently work independently to compute most of the volume, which lives at high-redshift. Only two communication steps are required for each tile: the LPT potentials are received at the beginning, and at the end of the computation each tile returns its final state at the redshift where it intersects the light-cone.

Node Failures. Robustness to node failure is an important consideration on all very large computational platforms. Even with extremely low failure probability for each node, since the number of nodes is high, the probability that some node fails during the course of a computation becomes high. After its initialisation steps (see Sect. 3.1), our approach is entirely robust to such failure, since any individual tile can be recomputed after the fact on a modest system, for very little cost.

5.2. Conclusion

In this paper, we introduced a perfectly parallel and easily applicable algorithm for cosmological simulations using sCOLA. Our approach is based on a tiling of the full simulation box, where each tile is run independently. By the use of buffer regions and appropriate Dirichlet boundary conditions, we improved the accuracy of the algorithm with respect to Tassev et al. (2015). In particular, we showed that suitable setups can reach 3% to 1% accuracy at all the scales simulated, as required for data analysis of the next generation of large-scale structure surveys. In case studies, we tested the relative impact of the two approximations involved in our approach, for density assignment and the boundary gravitational potential. We considered the computational cost of our algorithm and demonstrated that even if the CPU time needed is unavoidably higher, the wall-clock time and memory footprint can be drastically reduced.

This study opens up a wide range of possible extensions, discussed in Sect. 5.1. Benefiting from its perfect scalability, the approach could also allow for novel analyses of cosmological data from fully non-linear models previously too expensive to be tractable. It could straightforwardly be used for the construction of mock catalogues, but also within recently introduced likelihood-free inference techniques such as DELFI (Alsing et al. 2018), BOLFI (Leclercq 2018) and SELFI (Leclercq et al. 2019), which have a need for cheap simulator-based data models. We therefore anticipate that sCOLA will become an important tool in computational cosmology for the coming era.

Our perfectly parallel sCOLA algorithm has been implemented in the publicly available SIMBELMYNË code[10], where it is included in version 0.4.0 and later.

Acknowledgements. We are grateful to Matías Zaldarriaga for stimulating discussions and useful comments throughout the realisation of this project. We thank Jens Jasche and Svetlin Tassev for discussions that triggered this project, and Oliver Hahn for constructive observations. FL and BDW acknowledge the hospitality of the Institute for Advanced Study, Princeton, where this project was initiated. FL, BF and WJP thank the Institute of Cosmology and Gravitation of the University of Portsmouth, where part of this work was prepared. This work made use of NumPy (van der Walt et al. 2011), IPython (Perez & Granger 2007), Matplotlib (Hunter 2007), Jupyter notebooks (Kluyver et al. 2016), and the colourmaps provided by the cmocean (Thyng et al. 2016,) and CMasher (`https://github.com/1313e/CMasher`) packages. FL acknowledges funding from the Imperial College London Research Fellowship Scheme. GL and BDW acknowledge financial support from the ANR BIG4, under reference ANR-16-CE23-0002. The Center for Computational Astrophysics is supported by the Simons Foundation. Research at Perimeter Institute is supported in part by the Government of Canada through the Department of Innovation, Science and Economic Development Canada and by the Province of Ontario through the Ministry of Colleges and Universities. Numerical computations were done on the Sciama High Performance Compute (HPC) cluster which is supported by the ICG, SEPNet and the University of Portsmouth; and on the cx1 cluster hosted by the Research Computing Service facilities at Imperial College London (`doi:10.14469/hpc/2232`). This work was done within the Aquila Consortium (`https://aquila-consortium.org`). FL and BDW conceived the project. FL wrote the SIMBELMYNË code, implemented the new sCOLA algorithm, ran the simulations, performed the study, supervised BF's internship project, and wrote the bulk of the paper. BF contributed to the first implementation of the tiling algorithm in SIMBELMYNË and to early tests of the method. GL suggested the investigation of an alternative Poisson solver, proposed tests of the impact of boundary effects, and contributed to writing the paper. BDW made conceptual contributions, helped designing the accuracy and speed tests, and contributed to writing the paper. AHJ prompted the use of the linearly-evolving potential approximation. AHJ and AFH contributed to the interpretation of results. WJP supported the design of the first version of the algorithm and contributed to student supervision. CN contributed to the collegial construction of the standards of science, by developing the methodological framework, the state-of-the-art, as well as post-publication procedures. All natural authors read and approved the final manuscript.

References

Alimi, J. M., Bouillot, V., & Rasera, Y. 2012, DEUS Full Observable ΛCDM Universe Simulation: the numerical challenge
Alsing, J., Wandelt, B., & Feeney, S. 2018, MNRAS, 477, 2874
Amdahl, G. M. 1967, in Proceedings of the April 18–20, 1967, Spring Joint Computer Conference, AFIPS '67 (Spring) (New York, NY, USA: Association for Computing Machinery), 483
Aubert, D., Deparis, N., & Ocvirk, P. 2015, MNRAS, 454, 1012
Audren, B., Lesgourgues, J., Bird, S., Haehnelt, M. G., & Viel, M. 2013, J. Cosmology Astropart. Phys., 2013, 026
Austermann, J. E., Aird, K. A., & Beall, J. A. 2012, SPTpol: an instrument for CMB polarization measurements with the South Pole Telescope, SPIE Conf. Ser., 8452, 84521E
Bagla, J. S., & Padmanabhan, T. 1994, MNRAS, 266, 227
Barausse, E., Berti, E., & Hertog, T. 2020, Prospects for Fundamental Physics with LISA
Berger, M. J., & Colella, P. 1989, J. Comp. Phys., 82, 64
Bernardeau, F., Colombi, S., Gaztañaga, E., & Scoccimarro, R. 2002, Phys. Rep., 367, 1

[9] `https://www.cosmologyathome.org/`

[10] `https://bitbucket.org/florent-leclercq/simbelmyne/`

Birdsall, C. K., & Langdon, A. B. 1985, Plasma Physics via Computer Simulation (CRC Press)
Bouchet, F. R., Colombi, S., Hivon, E., & Juszkiewicz, R. 1995, A&A, 296, 575
Brainerd, T. G., Scherrer, R. J., & Villumsen, J. V. 1993, ApJ, 418, 570
Bryan, G. L., Norman, M. L., O'Shea, B. W., et al. 2014, ApJS, 211, 19
Buchert, T., Melott, A. L., & Weiß, A. G. 1994, A&A, 288, 349
Cheng, S., Yu, H.-R., Inman, D., et al. 2020, CUBE - Towards an Optimal Scaling of Cosmological N-body Simulations
Chisari, N. E., Richardson, M. L. A., Devriendt, J., et al. 2018, MNRAS, 480, 3962
DESI Collaboration 2016, The DESI Experiment Part I: Science, Targeting, and Survey Design
Eisenstein, D. J., & Hu, W. 1998, ApJ, 496, 605
Eisenstein, D. J., & Hu, W. 1999, ApJ, 511, 5
Frigo, M., & Johnson, S. G. 2005, Program Generation, Optimization, and Platform Adaptation, Proc. IEEE, 93, 216
Fryxell, B., Olson, K., Ricker, P., et al. 2000, ApJS, 131, 273
Garrison, L. H., Eisenstein, D. J., & Pinto, P. A. 2019, MNRAS, 485, 3370
Gonnet, P., Schaller, M., Theuns, T., & Chalk, A. B. G. 2013, SWIFT: Fast Algorithms for Multi-resolution SPH on Multi-Core Architectures
Guillet, T., & Teyssier, R. 2011, J. Comp. Phys., 230, 4756
Hahn, O., & Abel, T. 2011, MNRAS, 415, 2101
Hahn, O., Angulo, R. E., & Abel, T. 2015, MNRAS, 454, 3920
Hockney, R. W., & Eastwood, J. W. 1981, Computer Simulation Using Particles (McGraw-Hill)
Howlett, C., Manera, M., & Percival, W. J. 2015, Astron. Comput., 12, 109
Hunter, J. D. 2007, Comput. Sci. Eng., 9, 90
Huterer, D., & Takada, M. 2005, Astroparticle Phys., 23, 369
Izard, A., Crocce, M., & Fosalba, P. 2016, MNRAS, 459, 2327
James, R. A. 1977, J. Comp. Phys., 25, 71
Johnson, S. G., & Frigo, M. 2008, in Fast Fourier Transforms, ed. C. S. Burrus (Houston TX: Rice University Connexions)
Kluyver, T., Ragan-Kelley, B., & Pérez, F. 2016, ELPUB
Klypin, A., & Holtzman, J. 1997, Particle-Mesh code for cosmological simulations
Knebe, A., & Doumler, T. 2010, AMIGA: Adaptive Mesh Investigations of Galaxy Assembly
Koda, J., Blake, C., Beutler, F., Kazin, E., & Marin, F. 2016, MNRAS, 459, 2118
Laureijs, R., Amiaux, J., & Arduini, S. 2011, Euclid Definition Study Report
Leclercq, F. 2015, Ph.D. Thesis, Institut d'Astrophysique de Paris
Leclercq, F. 2018, Phys. Rev. D, 98, 063511
Leclercq, F., Jasche, J., & Wandelt, B. 2015, J. Cosmol. Astropart. Phys., 6, 15
Leclercq, F., Jasche, J., Lavaux, G., Wandelt, B., & Percival, W. 2017, J. Cosmol. Astropart. Phys., 6, 049
Leclercq, F., Enzi, W., Jasche, J., & Heavens, A. 2019, MNRAS, 490, 4237
LIGO Scientific Collaboration 2015, Class. Quant. Grav., 32, 074001
LSST Science Collaboration 2012, Large Synoptic Survey Telescope: Dark Energy Science Collaboration
Merloni, A., Predehl, P., Becker, W., et al. 2012, eROSITA Science Book: Mapping the Structure of the (Energetic Universe)
Ocvirk, P., Gillet, N., Shapiro, P. R., et al. 2016, MNRAS, 463, 1462
Perez, F., & Granger, B. E. 2007, Comput. Sci. Eng., 9, 21
Planck Collaboration XIII. 2016, A&A, 594, A13
Potter, D., Stadel, J., & Teyssier, R. 2017, Comput Astrophys. Cosmology., 4, 2
Quinn, T., Katz, N., Stadel, J., & Lake, G. 1997, Time stepping N-body simulations
Schneider, A., Teyssier, R., Stadel, J., et al. 2019, J. Cosmol. Astropart. Phys., 3, 020
Simon, S. M., Beall, J. A., Cothard, N. F., et al. 2018, J. Low. Temp. Phys., 193, 1041
Simons Observatory Collaboration 2019, J. Cosmol. Astropart. Phys., 2, 056
SPHEREx Science Team 2018, Am. Astron. Soc. Meet. Abstr., 231, 354.21
Square Kilometre Array Cosmology Science Working Group 2018, Cosmology with Phase 1 of the Square Kilometre Array; Red Book 2018: Technical specifications and performance forecasts
Strauss, M. A., Cen, R., Ostriker, J. P., Lauer, T. R., & Postman, M. 1995, ApJ, 444, 507
Tassev, S., & Zaldarriaga, M. 2012, J. Cosmol. Astropart. Phys., 12, 11
Tassev, S., Eisenstein, D. J., Wandelt, B. D., & Zaldarriaga, M. 2015, sCOLA: The N-body COLA Method Extended to the Spatial Domain
Tassev, S., Zaldarriaga, M., & Eisenstein, D. J. 2013, J. Cosmol. Astropart. Phys., 6, 36
Teyssier, R. 2002, A&A, 385, 337
The Virgo Collaboration 2020, J. Phys. Conf. Ser., 1342, 012010
Theuns, T., Chalk, A., Schaller, M., & Gonnet, P. 2015, SWIFT: task-based Hydrodynamics and Gravity for Cosmological Simulations
Thyng, K. M., Greene, C. A., Hetland, R. D., Zimmerle, H. M. 2016, Oceanography, 29
van Daalen, M. P., Schaye, J., Booth, C. M., & Dalla Vecchia, C. 2011, MNRAS, 415, 3649
van der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, Comput. Sci. Eng., 13, 22
Yu, H.-R., Pen, U.-L., & Wang, X. 2018, ApJS, 237, 24
Zel'dovich, Y. B. 1970, A&A, 5, 84

Appendix A: Model equations

A.1. Model equations in the standard PM code

Denoting by a the scale factor of the Universe and τ the conformal time, a PM code solves the equations of motion for the position $\boldsymbol{x}$ and momentum $\boldsymbol{p}$ of dark matter particles in comoving coordinates (the mass of particles m is absorbed in the definition of the momentum $\boldsymbol{p}$):

$$\boldsymbol{p} = a\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\tau}, \tag{A.1}$$

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}\tau} = -a\boldsymbol{\nabla}_{\boldsymbol{x}}\Phi(\boldsymbol{x},\tau), \tag{A.2}$$

coupled to the Poisson equation for the gravitational potential, sourced by density fluctuations (Eq. (8)),

$$\Delta_{\boldsymbol{x}}\Phi(\boldsymbol{x},\tau) = 4\pi \mathrm{G}a^2\bar{\rho}(\tau)\delta(\boldsymbol{x},\tau), \tag{A.3}$$

where G is the gravitational constant and $\bar{\rho}(\tau)$ is the mean matter density at conformal time τ. The density contrast is defined from the local matter density $\rho(\boldsymbol{x},\tau)$ by

$$\delta(\boldsymbol{x},\tau) \equiv \frac{\rho(\boldsymbol{x},\tau)}{\bar{\rho}(\tau)} - 1. \tag{A.4}$$

For simplicity, from now on we note $\boldsymbol{\nabla}_{\boldsymbol{x}} = \boldsymbol{\nabla}$, $\Delta_{\boldsymbol{x}} = \Delta$ and $\delta(\boldsymbol{x},\tau) = \delta$.

It is convenient to choose the scale factor as time variable. Using $\partial_\tau = a'\,\partial_a$ and the background evolution $\bar{\rho}(\tau) = \rho^{(0)}a^{-3}$ (a prime denotes a differentiation with respect to τ and the superscript (0) denotes quantities at the present time), the equations to solve are rewritten:

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}a} = \frac{\boldsymbol{p}}{a'a}, \tag{A.5}$$

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}a} = -\frac{a}{a'}\boldsymbol{\nabla}\Phi, \tag{A.6}$$

$$\Delta\Phi = 4\pi\mathrm{G}\rho^{(0)}a^{-1}\delta \equiv \frac{3}{2}\Omega_{\mathrm{m}}^{(0)}a^{-1}\delta. \tag{A.7}$$

We will use the equivalent formulation

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}a} = \mathcal{D}(a)\boldsymbol{p} \quad \text{with} \quad \mathcal{D}(a) \equiv \frac{1}{a^2\mathcal{H}(a)}, \tag{A.8}$$

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}a} = \mathcal{K}(a)\boldsymbol{\nabla}\left(\Delta^{-1}\delta\right) \quad \text{with} \quad \mathcal{K}(a) \equiv -\frac{3}{2}\frac{\Omega_{\mathrm{m}}^{(0)}}{a\mathcal{H}(a)}, \tag{A.9}$$

where we have combined Eqs. (A.6) and (A.7), introduced the conformal Hubble factor $\mathcal{H}(a) \equiv a'/a$, and defined the 'drift prefactor' $\mathcal{D}(a)$ and the "kick prefactor" $\mathcal{K}(a)$.

A.2. Model equations with COLA

We now introduce the COLA scheme, following Tassev et al. (2013, 2015). For each particle, we work in the frame comoving with its LPT observer, whose position is given by (see Sect. 2.1)

$$\boldsymbol{x}_{\mathrm{LPT}}(a) = \boldsymbol{q} - D_1(a)\boldsymbol{\Psi}_1 + D_2(a)\boldsymbol{\Psi}_2, \tag{A.10}$$

where we have introduced the time-independent vectors $\boldsymbol{\Psi}_1 \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}\phi^{(1)}$ and $\boldsymbol{\Psi}_2 \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}\phi^{(2)}$. Noting $\boldsymbol{x}(a) = \boldsymbol{x}_{\mathrm{LPT}}(a) + \boldsymbol{x}_{\mathrm{res}}(a)$ the final position of the same particle, we have

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}a} = \frac{\mathrm{d}\boldsymbol{x}_{\mathrm{LPT}}}{\mathrm{d}a} + \frac{\mathrm{d}\boldsymbol{x}_{\mathrm{res}}}{\mathrm{d}a}$$

with

$$\frac{\mathrm{d}\boldsymbol{x}_{\mathrm{LPT}}}{\mathrm{d}a} = -\frac{\mathrm{d}D_1}{\mathrm{d}a}\boldsymbol{\Psi}_1 + \frac{\mathrm{d}D_2}{\mathrm{d}a}\boldsymbol{\Psi}_2 \equiv \mathcal{D}(a)\boldsymbol{p}_{\mathrm{LPT}}. \tag{A.11}$$

We also define $\boldsymbol{p}_{\mathrm{res}}$ such that $\mathrm{d}\boldsymbol{x}_{\mathrm{res}}/\mathrm{d}a \equiv \mathcal{D}(a)\boldsymbol{p}_{\mathrm{res}}$. Then $\boldsymbol{p} = \boldsymbol{p}_{\mathrm{LPT}} + \boldsymbol{p}_{\mathrm{MC}}$ (see Eq. (A.8)). Furthermore,

$$\frac{\mathrm{d}\boldsymbol{p}_{\mathrm{LPT}}}{\mathrm{d}a} = \frac{\mathrm{d}}{\mathrm{d}a}\left(\frac{1}{\mathcal{D}(a)}\frac{\mathrm{d}\boldsymbol{x}_{\mathrm{LPT}}}{\mathrm{d}a}\right) \equiv -\mathcal{K}(a)\mathcal{V}[\boldsymbol{x}_{\mathrm{LPT}}](a), \tag{A.12}$$

where the differential operator $\mathcal{V}[\cdot](a)$ is defined by

$$\mathcal{V}[\cdot](a) \equiv -\frac{1}{\mathcal{K}(a)}\frac{\mathrm{d}}{\mathrm{d}a}\left(\frac{1}{\mathcal{D}(a)}\frac{\mathrm{d}\cdot}{\mathrm{d}a}\right). \tag{A.13}$$

With these notations, Eq. (A.9) reads

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}a} &= \frac{\mathrm{d}\boldsymbol{p}_{\mathrm{LPT}}}{\mathrm{d}a} + \frac{\mathrm{d}\boldsymbol{p}_{\mathrm{res}}}{\mathrm{d}a} \\ &= -\mathcal{K}(a)\mathcal{V}[\boldsymbol{x}_{\mathrm{LPT}}](a) + \frac{\mathrm{d}\boldsymbol{p}_{\mathrm{res}}}{\mathrm{d}a} = \mathcal{K}(a)\boldsymbol{\nabla}\left(\Delta^{-1}\delta\right).\end{aligned} \tag{A.14}$$

In COLA, the natural variables are therefore $\boldsymbol{x}$ and $\boldsymbol{p}_{\mathrm{res}}$.

As mentioned in Sect. 2.1, the key point in COLA is that the fictitious LPT force acting on particles, $\mathcal{V}[\boldsymbol{x}_{\mathrm{LPT}}](a)$, can be computed analytically. From Eq. (A.10), it is straightforward to check that $\mathcal{V}[\boldsymbol{x}_{\mathrm{LPT}}](a) = -\mathcal{V}[D_1](a)\boldsymbol{\Psi}_1 + \mathcal{V}[D_2](a)\boldsymbol{\Psi}_2$. The computation of $\mathcal{V}[D_1](a)$ and $\mathcal{V}[D_2](a)$ uses the differential equations followed by the linear and second-order growth factor, as well as the second Friedmann equation. The result is (see e.g. Leclercq 2015, Eqs. (1.7), (1.96), (1.118) and Appendix B)

$$\mathcal{V}[D_1](a) = D_1(a), \tag{A.15}$$

$$\mathcal{V}[D_2](a) = D_2(a) - D_1^2(a). \tag{A.16}$$

The equations of motion to solve are therefore, in tCOLA,

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}a} = \mathcal{D}(a)\boldsymbol{p}_{\mathrm{res}} - \frac{\mathrm{d}D_1}{\mathrm{d}a}\boldsymbol{\Psi}_1 + \frac{\mathrm{d}D_2}{\mathrm{d}a}\boldsymbol{\Psi}_2, \tag{A.17}$$

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{p}_{\mathrm{res}}}{\mathrm{d}a} = \mathcal{K}(a)\Big[&\boldsymbol{\nabla}\left(\Delta^{-1}\delta\right) - D_1(a)\boldsymbol{\Psi}_1 \\ &+ \left(D_2(a) - D_1^2(a)\right)\boldsymbol{\Psi}_2\Big].\end{aligned} \tag{A.18}$$

These are mathematically equivalent to the equations of motion of a PM code (Eqs. (A.8) and (A.9)). In sCOLA, the "kick equation" (Eq. (A.18)) is replaced for each particle of the sCOLA box by the approximation (Tassev et al. 2015)

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{p}_{\mathrm{res}}}{\mathrm{d}a} = \mathcal{K}(a)\Big[&\boldsymbol{\nabla}^{\mathrm{sCOLA}}\left((\Delta^{\mathrm{sCOLA}})^{-1}\delta^{\mathrm{sCOLA}}\right) - D_1(a)\boldsymbol{\Psi}_1^{\mathrm{sCOLA}} \\ &+ \left(D_2(a) - D_1^2(a)\right)\boldsymbol{\Psi}_2^{\mathrm{sCOLA}}\Big].\end{aligned} \tag{A.19}$$

with the notations introduced in Sect. 2.3, as well as $\boldsymbol{\Psi}_1^{\mathrm{sCOLA}} \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}^{\mathrm{sCOLA}}\phi^{(1)}$ and $\boldsymbol{\Psi}_2^{\mathrm{sCOLA}} \equiv \boldsymbol{\nabla}_{\boldsymbol{q}}^{\mathrm{sCOLA}}\phi^{(2)}$. Importantly, the "drift equation" (Eq. (A.17)) is not modified, since we are always, by definition, computing a residual displacement with respect to the LPT observer of the full box, whose position is given by Eq. (A.10).

Appendix B: Standard and modified time-stepping

B.1. Time-stepping in the standard PM algorithm

In this paper, we adopt the second-order symplectic "kick-drift-kick" algorithm, also known as the leapfrog scheme (e.g.

Birdsall & Langdon 1985) to integrate the equations of motion. This algorithm relies on integrating the equations on a small time-step and approximating the momenta ($\boldsymbol{p}$ in the "drift equation" (A.8)) and accelerations ($\boldsymbol{\nabla}(\Delta^{-1}\delta)$ in the "kick equation" (A.9)) that appear in the integrands by their value at some time within the interval. This defines the Drift (D) and Kick (K) operators, which read using the standard discretisation (Quinn et al. 1997):

$$\mathrm{D}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}}):$$
$$\boldsymbol{x}(t_i^{\mathrm{D}}) \mapsto \boldsymbol{x}(t_f^{\mathrm{D}}) = \boldsymbol{x}(t_i^{\mathrm{D}}) + \alpha_{\boldsymbol{p}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}})\boldsymbol{p}\left(t^{\mathrm{K}}\right), \tag{B.1}$$
$$\mathrm{K}(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}}):$$
$$\boldsymbol{p}(t_i^{\mathrm{D}}) \mapsto \boldsymbol{p}(t_f^{\mathrm{D}}) = \boldsymbol{p}(t_i^{\mathrm{D}}) + \beta_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}})\left[\boldsymbol{\nabla}\left(\Delta^{-1}\delta\right)\right](t^{\mathrm{D}}), \tag{B.2}$$

where

$$\alpha_{\boldsymbol{p}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}}) \equiv \int_{t_i^{\mathrm{D}}}^{t_f^{\mathrm{D}}} \mathcal{D}(\tilde{t})\,\mathrm{d}\tilde{t} = \int_{t_i^{\mathrm{D}}}^{t_f^{\mathrm{D}}} \frac{\mathrm{d}\tilde{t}}{\tilde{t}^2\mathcal{H}(\tilde{t})},$$
$$\beta_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}}) \equiv \int_{t_i^{\mathrm{K}}}^{t_f^{\mathrm{K}}} \mathcal{K}(\tilde{t})\,\mathrm{d}\tilde{t} = -\frac{3}{2}\Omega_{\mathrm{m}}^{(0)}\int_{t_i^{\mathrm{K}}}^{t_f^{\mathrm{K}}} \frac{\mathrm{d}\tilde{t}}{\tilde{t}\mathcal{H}(\tilde{t})}, \tag{B.3}$$

and t is a function of the scale factor a (typically $t(a) = a$ or $t(a) = \exp(a)$ for time-steps linearly spaced or logarithmically spaced in the scale factor, respectively).

The time evolution between $t_0 = t(a_{\mathrm{i}})$ and $t_{n+1} = t(a_{\mathrm{f}})$ is then achieved by applying the following operator, $\mathrm{E}(t_{n+1}, t_0)$, to the initial state $(\boldsymbol{x}(t_0), \boldsymbol{p}(t_0))$:

$$\mathrm{K}(t_{n+1/2}, t_{n+1}, t_{n+1})\mathrm{D}(t_n, t_{n+1}, t_{n+1/2}) \times \left[\prod_{i=0}^{n}\mathrm{K}(t_{i+1/2}, t_{i+3/2}, t_{i+1})\mathrm{D}(t_i, t_{i+1}, t_{i+1/2})\right]\mathrm{K}(t_0, t_{1/2}, t_0). \tag{B.4}$$

B.2. Time-stepping with COLA, standard discretisation

Using the standard discretisation (Quinn et al. 1997) of Eqs. (A.17) and (A.18), the Kick and Drift operators for tCOLA are defined by

$$\widetilde{\mathrm{D}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}}):$$
$$\boldsymbol{x}(t_i^{\mathrm{D}}) \mapsto \boldsymbol{x}(t_f^{\mathrm{D}}) = \boldsymbol{x}(t_i^{\mathrm{D}}) + \alpha_{\boldsymbol{p}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}})\boldsymbol{p}_{\mathrm{res}}\left(t^{\mathrm{K}}\right) - \left[D_1\right]_{t_i^{\mathrm{D}}}^{t_f^{\mathrm{D}}}\boldsymbol{\Psi}_1 + \left[D_2\right]_{t_i^{\mathrm{D}}}^{t_f^{\mathrm{D}}}\boldsymbol{\Psi}_2, \tag{B.5}$$
$$\widetilde{\mathrm{K}}(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}}):$$
$$\boldsymbol{p}_{\mathrm{res}}(t_i^{\mathrm{D}}) \mapsto \boldsymbol{p}_{\mathrm{res}}(t_f^{\mathrm{D}}) = \boldsymbol{p}_{\mathrm{res}}(t_i^{\mathrm{D}}) + \beta_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}}) \times \left(\left[\boldsymbol{\nabla}\left(\Delta^{-1}\delta\right)\right](t^{\mathrm{D}}) - D_1(t^{\mathrm{D}})\boldsymbol{\Psi}_1 + \left(D_2(t^{\mathrm{D}}) - D_1^2(t^{\mathrm{D}})\right)\boldsymbol{\Psi}_2\right), \tag{B.6}$$

where the time factors $\alpha_{\boldsymbol{p}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}})$ and $\beta_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}})$ are the same as in the PM case (see Eq. (B.3)). For sCOLA, $\widetilde{\mathrm{K}}$ is given by the same expression (Eq. (B.6)) but operates on quantities and differential operators superscripted "sCOLA" consistently with Eq. (A.19).

In the initial conditions, generated with LPT, we have $\boldsymbol{p} = \boldsymbol{p}_{\mathrm{LPT}}$; therefore the momentum residual in the rest frame of LPT observers, $\boldsymbol{p}_{\mathrm{res}}$, should be initialised to zero. At the end, the LPT momentum $\boldsymbol{p}_{\mathrm{LPT}}$ has to be added to $\boldsymbol{p}_{\mathrm{res}}$ to recover the full momentum of particles, $\boldsymbol{p}$. This corresponds respectively to the L_- and L_+ operators (Tassev et al. 2013, Appendix A), given by

$$\mathrm{L}_\pm(t): \quad \boldsymbol{p}(t) \mapsto \boldsymbol{p}(t) \pm \boldsymbol{p}_{\mathrm{LPT}}(t) = \boldsymbol{p}(t) \pm \frac{1}{\mathcal{D}(t)}\left(-\frac{\mathrm{d}D_1}{\mathrm{d}t}\boldsymbol{\Psi}_1 + \frac{\mathrm{d}D_2}{\mathrm{d}t}\boldsymbol{\Psi}_2\right). \tag{B.7}$$

In COLA, the time evolution between $t_0 = t(a_{\mathrm{i}})$ and $t_{n+1} = t(a_{\mathrm{f}})$ is therefore achieved by applying the following operator to the initial state $(\boldsymbol{x}(t_0), \boldsymbol{p}(t_0))$:

$$\mathrm{L}_+(t_{n+1})\widetilde{\mathrm{E}}(t_{n+1}, t_0)\mathrm{L}_-(t_0), \tag{B.8}$$

where $\widetilde{\mathrm{E}}(t_{n+1}, t_0)$ is the operator given by Eq. (B.4), replacing D by $\widetilde{\mathrm{D}}$ and K by $\widetilde{\mathrm{K}}$.

B.3. Time-stepping with COLA, modified discretisation

Another approach for the discretisation of Eqs. (A.17) and (A.18) is proposed by Tassev et al. (2013). For any arbitrary positive function u of t, we can rewrite

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \mathcal{D}(t)u(t)\left\{\frac{1}{u(t)} \times \boldsymbol{p}_{\mathrm{res}}\right\} - \frac{\mathrm{d}D_1}{\mathrm{d}t}\boldsymbol{\Psi}_1 + \frac{\mathrm{d}D_2}{\mathrm{d}t}\boldsymbol{\Psi}_2, \tag{B.9}$$
$$\frac{\mathrm{d}\boldsymbol{p}_{\mathrm{res}}}{\mathrm{d}t} = \frac{\mathrm{d}u(t)}{\mathrm{d}t}\left\{\frac{\mathcal{K}(t)}{\mathrm{d}u(t)/\mathrm{d}t} \times \left[\boldsymbol{\nabla}\left(\Delta^{-1}\delta\right) - D_1(t)\boldsymbol{\Psi}_1 + \left(D_2(t) - D_1^2(t)\right)\boldsymbol{\Psi}_2\right]\right\}. \tag{B.10}$$

This form is particularly relevant if $\boldsymbol{p}_{\mathrm{res}}$ has a time dependence which is entirely captured by a particular $u(t)$, which is universal for all particles. For each equation, considering that the part between curly brackets is constant during the time-step (instead of the momentum and accelerations, respectively), the modified $\widetilde{\mathrm{D}}$ and $\widetilde{\mathrm{K}}$ operators are given by Eqs. (B.5) and (B.6) with the following modified time factors instead of $\alpha_{\boldsymbol{p}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}})$ and $\beta_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}})$:

$$\tilde{\alpha}_{\boldsymbol{p}}(t_i^{\mathrm{D}}, t_f^{\mathrm{D}}, t^{\mathrm{K}}) \equiv \frac{1}{u(t^{\mathrm{K}})}\int_{t_i^{\mathrm{D}}}^{t_f^{\mathrm{D}}} \mathcal{D}(\tilde{t})u(\tilde{t})\,\mathrm{d}\tilde{t},$$
$$\tilde{\beta}_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}}) \equiv \left(u(t_f^{\mathrm{K}}) - u(t_i^{\mathrm{K}})\right) \times \frac{\mathcal{K}(t^{\mathrm{D}})}{(\mathrm{d}u(t)/\mathrm{d}t)(t^{\mathrm{D}})}, \tag{B.11}$$

where in $\tilde{\beta}_\delta(t_i^{\mathrm{K}}, t_f^{\mathrm{K}}, t^{\mathrm{D}})$, we have used the trivial integration $\int_{t_i^{\mathrm{K}}}^{t_f^{\mathrm{K}}} \frac{\mathrm{d}u(\tilde{t})}{\mathrm{d}\tilde{t}}\,\mathrm{d}\tilde{t} = u(t_f^{\mathrm{K}}) - u(t_i^{\mathrm{K}})$.

Using the Ansatz suggested by Tassev et al. (2013), $u(a) = a^{n_{\mathrm{LPT}}}$ when $t(a) = a$, we get the explicit expressions

$$\tilde{\alpha}_{\boldsymbol{p}}(a_i^{\mathrm{D}}, a_f^{\mathrm{D}}, a^{\mathrm{K}}) = \frac{1}{(a^{\mathrm{K}})^{n_{\mathrm{LPT}}}}\int_{a_i^{\mathrm{D}}}^{a_f^{\mathrm{D}}} \frac{\tilde{a}^{n_{\mathrm{LPT}}-2}}{\mathcal{H}(\tilde{a})}\,\mathrm{d}\tilde{a},$$
$$\tilde{\beta}_\delta(a_i^{\mathrm{K}}, a_f^{\mathrm{K}}, a^{\mathrm{D}}) = -\frac{3}{2}\Omega_{\mathrm{m}}^{(0)}\frac{(a_f^{\mathrm{K}})^{n_{\mathrm{LPT}}} - (a_i^{\mathrm{K}})^{n_{\mathrm{LPT}}}}{n_{\mathrm{LPT}}(a^{\mathrm{D}})^{n_{\mathrm{LPT}}}\mathcal{H}(a^{\mathrm{D}})}. \tag{B.12}$$

We adopt this form and $n_{\mathrm{LPT}} = -2.5$ for both tCOLA and sCOLA operators, throughout this paper.

Appendix C: Poisson solver with Dirichlet boundary conditions

In this appendix, we describe how to compute the interior gravitational potential Φ with Dirichlet boundary conditions. The

method is standard in computational physics and has been used at least since James (1977). Formally, we seek to solve the discrete Poisson equation,

$$\Delta\Phi = \delta, \tag{C.1}$$

subject to a known boundary potential $\Phi_{\rm BCs}$, where Δ is the FDA to the exact Laplacian operator, i.e. $\Delta \equiv \partial_x^2 + \partial_y^2 + \partial_z^2$ where ∂_x^2, ∂_y^2, and ∂_z^2 are discrete one-dimensional second-order derivatives (see Table 1 in Hahn & Abel 2011, for their expressions in FDA at order 2, 4 and 6).

C.1. Modified density distribution

We define $\Phi_{\rm BCs}$ as having non-zero values only in a layer of $N_{\rm ghost}$ cells immediately outside the active domain of the PM grid. We can then write the desired potential as $\Phi \equiv \widetilde{\Phi} + \Phi_{\rm BCs}$ where the required boundary condition for $\widetilde{\Phi}$ is $\widetilde{\Phi} = 0$. From the definition of $\Phi_{\rm BCs}$, $\Delta\Phi_{\rm BCs}$ will be non-zero only in a layer of $N_{\rm ghost}$ active cells just inside the domain boundaries. We can thus define a modified density distribution,

$$\delta' \equiv \delta - \Delta\Phi_{\rm BCs}, \tag{C.2}$$

which is the same as δ everywhere except in the layer of $N_{\rm ghost}$ cells adjoining the domain boundaries. We can then employ a zero-boundary condition Poisson solver to obtain a solution of $\Delta\widetilde{\Phi} = \delta'$ (see Sect. C.2). Within the interior, where $\Phi_{\rm BCs} = 0$, this solution is the desired final solution of $\Delta\Phi = \delta$ with the Dirichlet boundary condition $\Phi = \Phi_{\rm BCs}$.

C.2. Zero-boundary condition Poisson solver

In cosmological simulations, it is conventional to use FFTs to solve the Poisson equation, since the discrete Laplacian operator is diagonal in Fourier space. FFTs assume that the input source has periodic boundary conditions. Similarly, for zero boundary conditions, we can work with three-dimensional type-I discrete sine transforms (DST-I), defined by

$$\delta^{\ell,m,n} \equiv \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\sum_{k=1}^{N_z} \delta_{i,j,k} \mathcal{X}_i^\ell \mathcal{Y}_j^m \mathcal{Z}_k^n, \tag{C.3}$$

where $\delta_{i,j,k}$ is the value of the source field in the voxel indexed by $1 \le i \le N_x$, $1 \le y \le N_y$, $1 \le k \le N_z$ ($N_x = N_y = N_z = N_{\rm g}$ in this paper). The basis functions are defined by

$$\mathcal{X}_i^\ell \equiv \sin\left(\frac{\pi\ell i}{N_x+1}\right), \quad \mathcal{Y}_j^m \equiv \sin\left(\frac{\pi m j}{N_y+1}\right), \quad \mathcal{Z}_k^n \equiv \sin\left(\frac{\pi n k}{N_z+1}\right). \tag{C.4}$$

They ensure that the signal has zero boundary values (for $i \in \{0, N_x+1\}$ or $j \in \{0, N_y+1\}$ or $k \in \{0, N_z+1\}$). They satisfy discrete orthogonality relations, for example,

$$\frac{2}{N_x+1}\sum_{i=1}^{N_x} \mathcal{X}_i^\ell \mathcal{X}_i^{\ell'} = \delta_{\rm K}^{\ell\ell'} \quad \text{and} \quad \frac{2}{N_y+1}\sum_{m=1}^{N_y} \mathcal{Y}_j^m \mathcal{Y}_{j'}^m = \delta_{\rm K}^{jj'}, \tag{C.5}$$

where $\delta_{\rm K}$ is the Kronecker symbol. The inverse transformation is simply DST-I multiplied by $8/\left[(N_x+1)(N_y+1)(N_z+1)\right]$, i.e.

$$\delta_{i,j,k} = \frac{8}{(N_x+1)(N_y+1)(N_z+1)} \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\sum_{k=1}^{N_z} \delta^{\ell,m,n} \mathcal{X}_i^\ell \mathcal{Y}_j^m \mathcal{Z}_k^n, \tag{C.6}$$

and similarly for the gravitational potential,

$$\Phi_{i,j,k} = \frac{8}{(N_x+1)(N_y+1)(N_z+1)} \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\sum_{k=1}^{N_z} \Phi^{\ell,m,n} \mathcal{X}_i^\ell \mathcal{Y}_j^m \mathcal{Z}_k^n. \tag{C.7}$$

It is straightforward to show that $\mathcal{X}_i^\ell$, $\mathcal{Y}_j^m$, $\mathcal{Z}_k^n$ are eigenfunctions of the discrete one-dimensional second-order derivatives ∂_x^2, ∂_y^2, and ∂_z^2, respectively. The corresponding eigenvalues λ_x^ℓ, λ_y^m and λ_z^n are given by

$$\lambda_x^\ell = -\frac{4}{d_x^2}\sin^2\left(\frac{k_\ell d_x}{4}\frac{N_x}{N_x+1}\right), \tag{C.8}$$

$$\lambda_x^\ell = \frac{1}{3d_x^2}\left[\sin^2\left(\frac{k_\ell d_x}{2}\frac{N_x}{N_x+1}\right) - 16\sin^2\left(\frac{k_\ell d_x}{4}\frac{N_x}{N_x+1}\right)\right], \tag{C.9}$$

$$\lambda_x^\ell = -\frac{1}{45d_x^2}\left[2\sin^2\left(\frac{3k_\ell d_x}{4}\frac{N_x}{N_x+1}\right) - 27\sin^2\left(\frac{k_\ell d_x}{2}\frac{N_x}{N_x+1}\right) + 270\sin^2\left(\frac{k_\ell d_x}{4}\frac{N_x}{N_x+1}\right)\right], \tag{C.10}$$

for FDA at order 2, 4, and 6 respectively, where $k_\ell \equiv 2\pi\ell/L_x$, $d_x \equiv L_x/N_x$ and L_x is the size of the box along the x-direction ($L_x = L_{\rm sCOLA} \equiv L/N_{\rm tiles}$ in this paper). Similar expressions exist for λ_y^m and λ_z^n.

Plugging Eqs. (C.6) and (C.7) into (C.1) and using the orthogonality relations, we obtain a simple form for the discretised Poisson equation in sine space,

$$\Phi^{\ell,m,n} = \frac{\delta^{\ell,m,n}}{\lambda_x^\ell + \lambda_y^m + \lambda_z^n}. \tag{C.11}$$

Therefore, the Poisson equation $\Delta\Phi = \delta$ with zero boundary conditions can be solved by the following three steps:

1. performing a forward DST of the source ($\delta_{i,j,k} \rightarrow \delta^{\ell,m,n}$), according to Eq. (C.3) (costing $O(N_xN_yN_z \log\left[N_xN_yN_z\right])$ operations),
2. solving the Poisson equation in sine space ($\delta^{\ell,m,n} \rightarrow \Phi^{\ell,m,n}$), according to Eq. (C.11) (costing $O(N_xN_yN_z)$ operations),
3. performing an inverse DST of the gravitational potential ($\Phi^{\ell,m,n} \rightarrow \Phi_{i,j,k}$), according to Eq. (C.7) (costing $O(N_xN_yN_z \log\left[N_xN_yN_z\right])$ operations).

In practice, forward and inverse DSTs are performed using the FFTW library (Frigo & Johnson 2005), publicly available[11], where the DST-I is known as `FFTW_RODFT00`.

[11] http://www.fftw.org