Perfectly parallel cosmological simulations using spatial comoving Lagrangian acceleration

Florent Leclercq

www.florent-leclercq.eu

Imperial Centre for Inference and Cosmology
Imperial College London

Baptiste Faure, Guilhem Lavaux, Benjamin Wandelt,
Andrew Jaffe, Alan Heavens,
Will Percival, Camille Noûs

and the Aquila Consortium
www.aquila-consortium.org

arXiv:2003.04925

6 April 2020

ICIC
Imperial Centre
for Inference & Cosmology

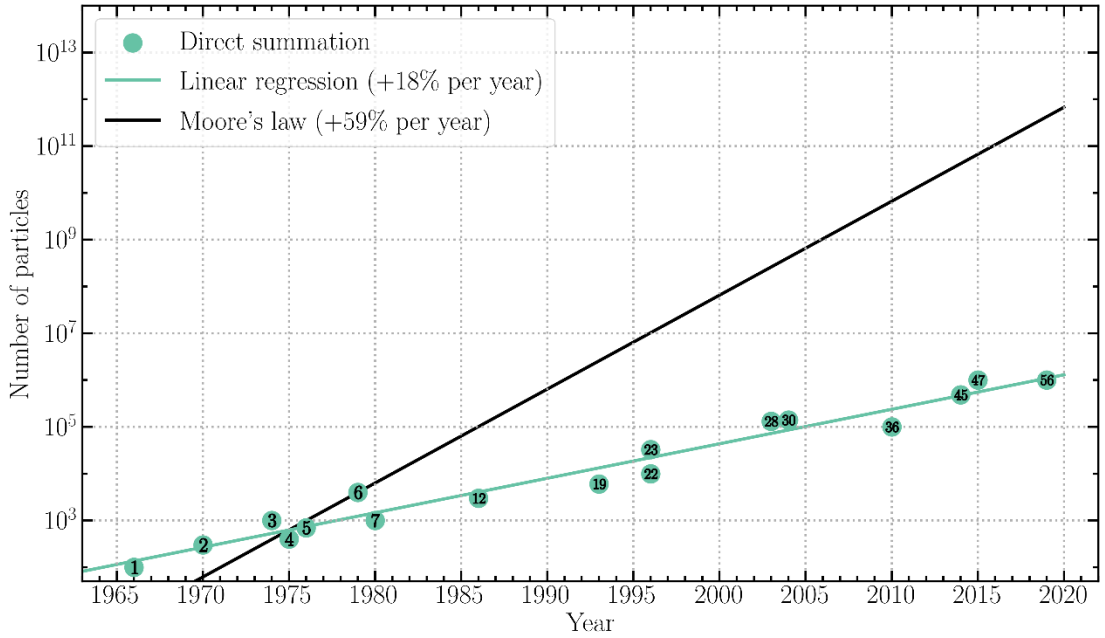
Imperial College
London

Why do we (still) need *N*-body codes?

- *N*-body simulations remain a basic ingredient for many cosmological modelling problems: galaxy clustering, ray-tracing, 21cm intensity mapping, Lyman-α
- Frequentist approach: mock surveys (e.g. DESI, Euclid, LSST) are used for measurements of summaries and their covariances
- Bayesian approach: forward numerical data models are the new way to express the theory
  - … embedded into a map-based likelihood: Bayesian large-scale structure inference (BORG)
  - … or in a simulator-based approach: likelihood-free inference (ABC, DELFI, BOLFI, SELFI)

Parallelisation of *N*-body codes: the challenge

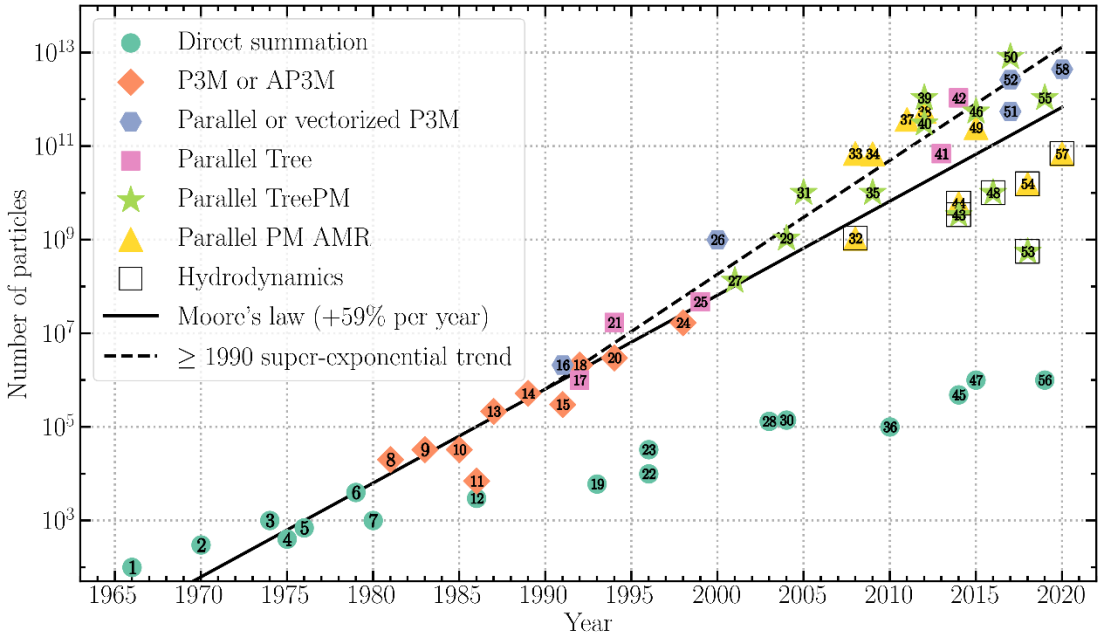
- The main issue preventing the easy parallelisation of *N*-body codes is the long-range nature of gravitational interactions
- "Exact" gravity requires $\mathcal{O}(N^2)$ all-to-all communications between *N* particles across the full computational volume.
- As a consequence "direct summation" simulations are unable to follow Moore's law for CPUs (doubling every 18 months)



References for all points available at http://florent-leclercq.eu/blog.php?page=2

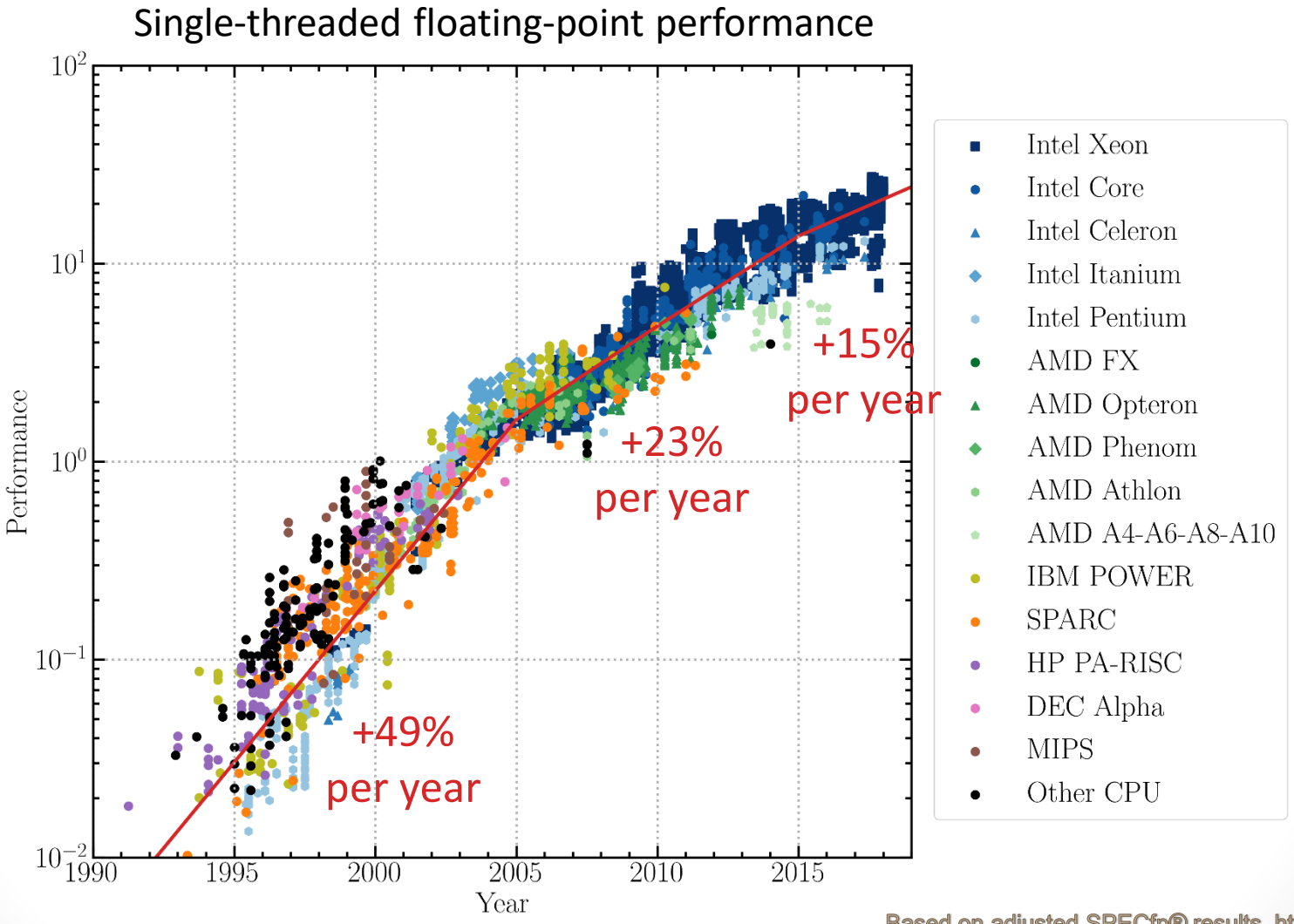
Parallelisation of N-body codes: the challenge

- Most of the work on numerical cosmology so far has focused on algorithms (such as tree, multipole, and mesh methods) that reduce the need for communications across the full computational volume
- Since 1990, a super-exponential trend that cannot be explained only by increase in computer speed can be observed



References for all points available at http://florent-leclercq.eu/blog.php?page=2

But per-core compute performance is slowing down



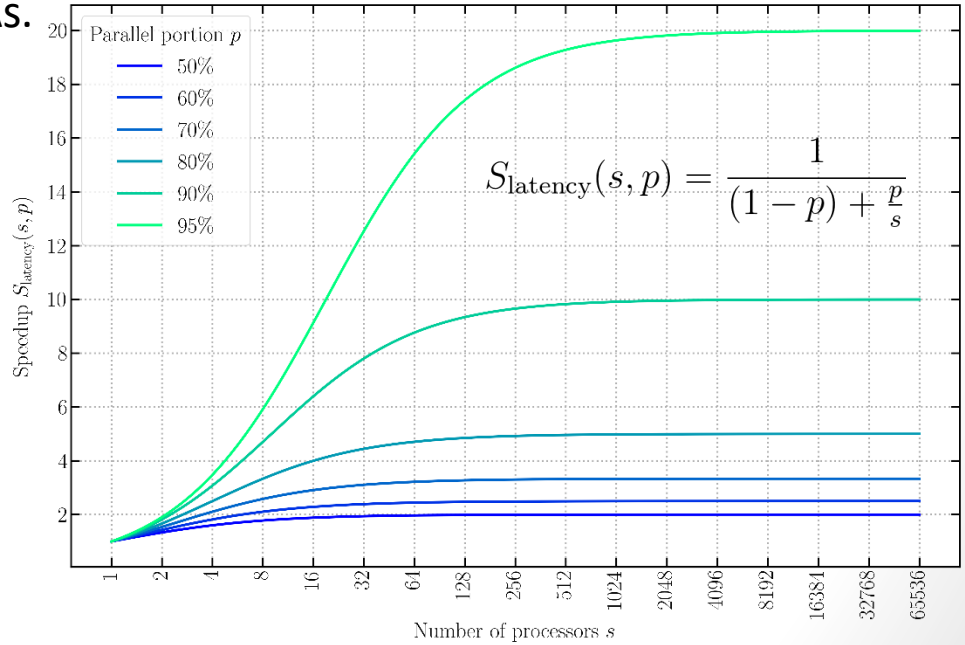
Based on adjusted SPECfp® results, http://spec.org

Cosmological simulations in the exascale world

- Traditional hardware architectures are reaching their physical limit.
- Current hardware development focuses on:
  - Packing a larger number of cores into each CPU: currently $\mathcal{O}(10^5)$, soon $\mathcal{O}(10^{6-7})$ in systems that are currently being built.
  - Developing hybrid architectures with cores + accelerators: GPUs and reconfigurable chips such as FPGAs.

- Compute cycles are no longer the scarce resource. The cost is driven by interconnections.
- Amdahl's law: latency kills the gains of parallelisation

Amdahl 1967, doi:10.1145/1465482.1465560

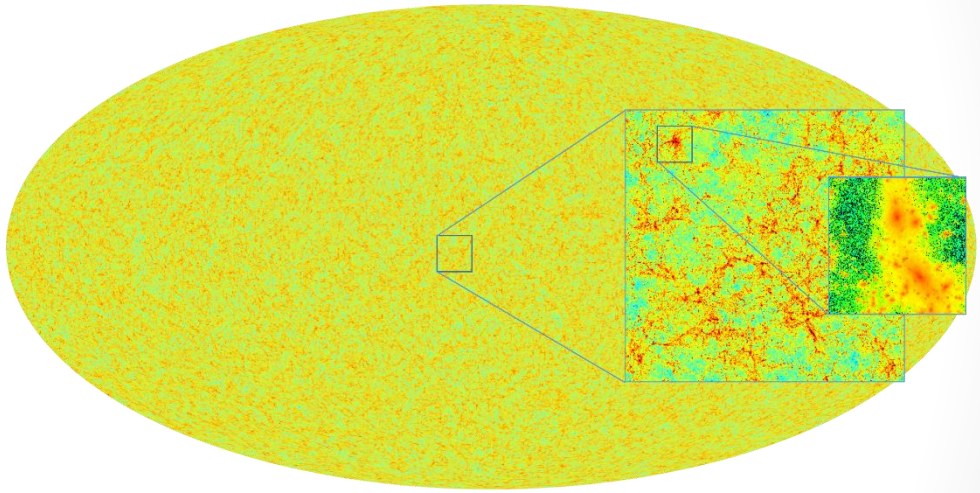


Cosmological simulations cannot merely rely on computers becoming faster to reduce the computational time.

The state of the art: beyond trillion particle simulations

A challenging problem:

- PKDGRAV3 code, 2 to 8 trillion particles, 4000+ GPU nodes, 350,000 node-hours
- Communication-dominated problem: nodes are tightly coupled with high-performance networks on the Titan supercomputer (100 M$).
- Sophisticated management is needed to maximise locality in the storage hierarchy (cache, RAM, disk I/O).

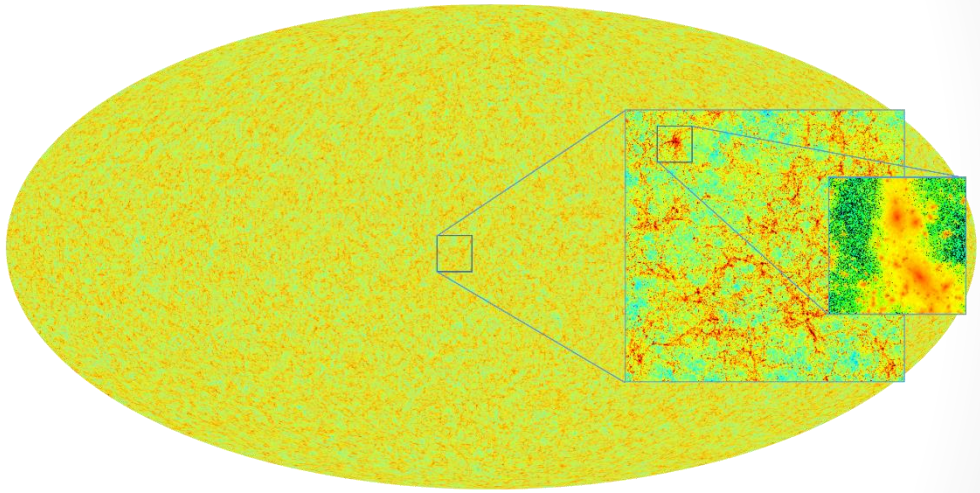


Potter, Stadel & Teyssier 2017, 1609.08621

The state of the art: beyond trillion particle simulations

~~A challenging problem~~: Shouldn't this be a simpler problem?

- PKDGRAV3 code, 2 to 8 trillion particles, 4000+ GPU nodes, 350,000 node-hours to solve quasi-linear evolution!
- Communication-dominated problem: nodes are tightly coupled with high-performance networks on the Titan supercomputer (100 M$). Are they really needed?
- Sophisticated management is needed to maximise locality in the storage hierarchy (cache, RAM, disk I/O). What if locality was ensured by construction?



Potter, Stadel & Teyssier 2017, 1609.08621

tCOLA: *Comoving Lagrangian Acceleration* (temporal domain)

- Write the displacement vector as: $\boldsymbol{\Psi} = \boldsymbol{\Psi}_{\rm LPT} + \boldsymbol{\Psi}_{\rm res} \quad (\mathbf{x} = \mathbf{q} + \boldsymbol{\Psi})$

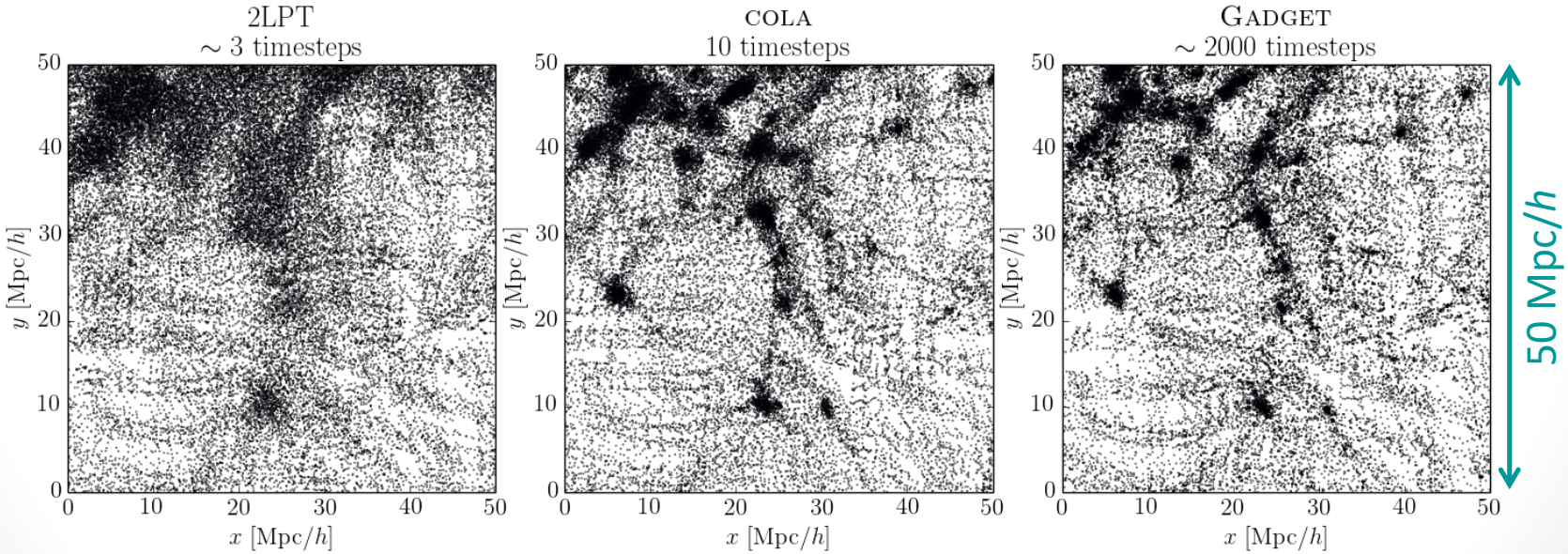
Tassev & Zaldarriaga 2012, 1203.5785

Analytical solutions!

- Time-stepping (omitted constants and Hubble expansion):

Standard: $\partial_a^2 \boldsymbol{\Psi} = -\boldsymbol{\nabla}_{\mathbf{x}} \Phi$ ⇒ **Modified**: $\partial_a^2 \boldsymbol{\Psi}_{\rm res} = \partial_a^2 (\boldsymbol{\Psi} - \boldsymbol{\Psi}_{\rm LPT}) = -\boldsymbol{\nabla}_{\mathbf{x}} \Phi - \partial_a^2 \boldsymbol{\Psi}_{\rm LPT}$

Tassev, Zaldarriaga & Einsenstein 2013, 1301.0322

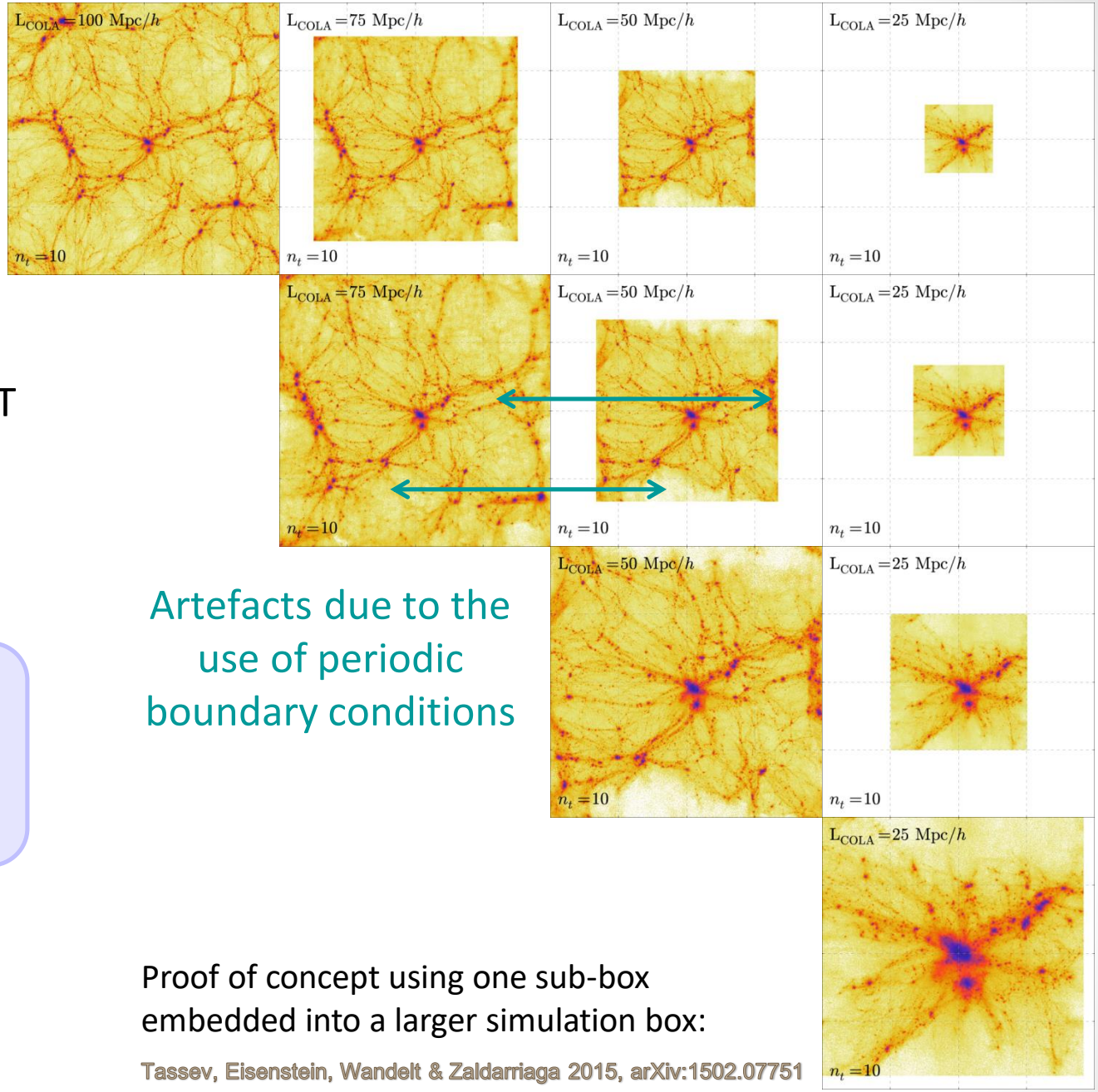


Beneficial gain of efficiency… but the real problem is not CPU-hours, but the inability to run on a very large number of cores due to latencies/parallelisation overhead.

sCOLA: Extension to the spatial domain

- Computing the LPT reference frame suggests a new strategy:

Can we decouple sub-volumes by using the large-scale analytical solution?



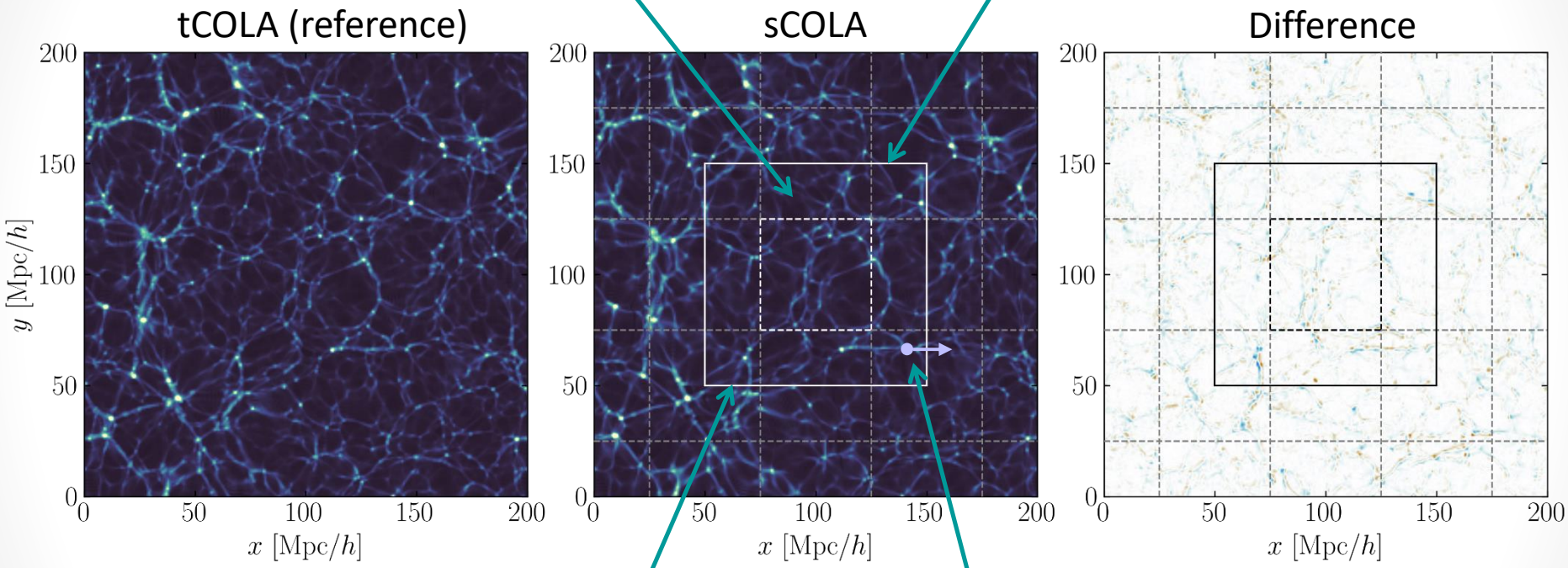
Artefacts due to the use of periodic boundary conditions

Proof of concept using one sub-box embedded into a larger simulation box:

Tassev, Eisenstein, Wandelt & Zaldarriaga 2015, arXiv:1502.07751

The solution to boundary artefacts

1. A buffer region around each tile
2. Appropriate Dirichlet boundary conditions for the potential



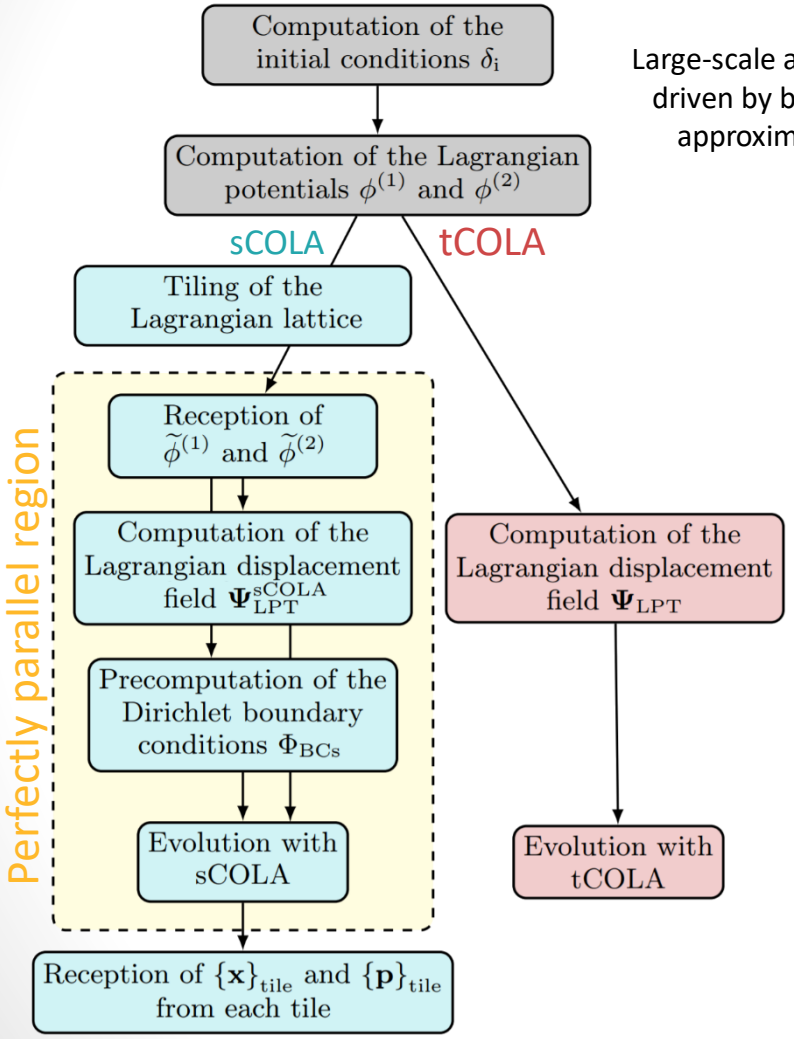
The Poisson solver uses discrete sine transforms (DSTs) instead of FFTs.

Two remaining approximations (to ensure no communication between tiles):

1. Linearly-evolving potential (LEP) at the boundaries: $\Phi_{\rm BCs}(\mathbf{x}, a) \approx D_1(a)\phi^{(1)}(\mathbf{x})$
2. Outgoing particles do not deposit mass

FL, Faure, Lavaux, Wandelt, Jaffe, Heavens, Percival & Noûs, 2003.04925

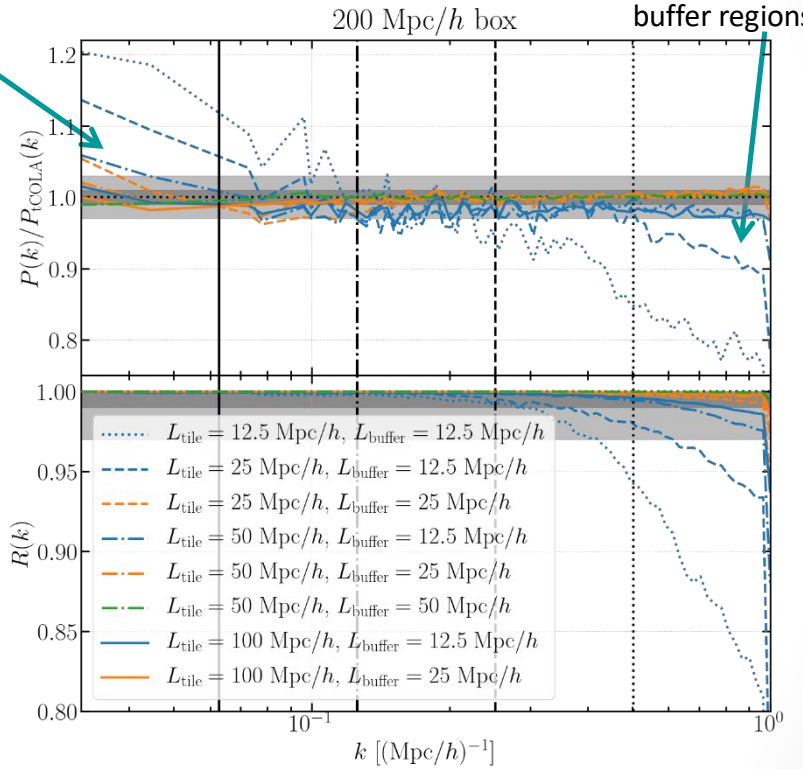
The perfectly parallel algorithm and its accuracy



- Parameter investigation (size of tiles and buffer regions):

Large-scale accuracy is driven by boundary approximations

Small-scale accuracy is driven by the size of buffer regions



- Some setups reach 3 to 1% accuracy at all scales, as required for the next-generation of surveys

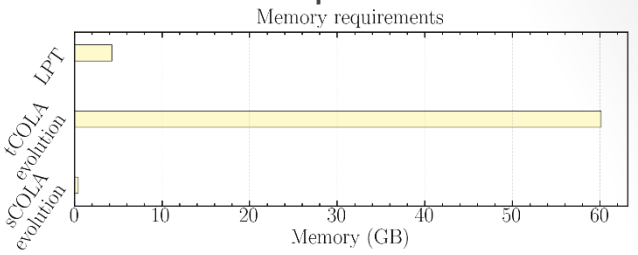
FL, Faure, Lavaux, Wandelt, Jaffe, Heavens, Percival & Noûs, 2003.04925

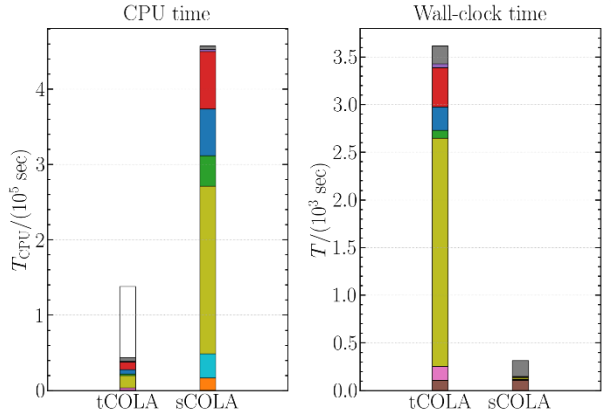
Memory requirements, parallelisation potential & speed

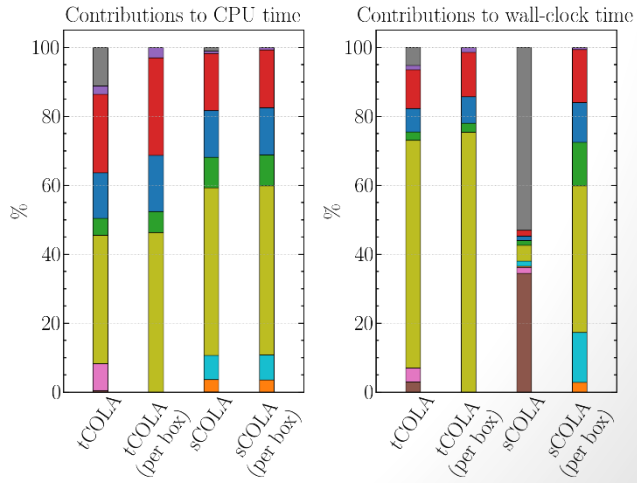
- Buffer regions require to oversimulate the volume by a factor r
- But small *N*-body simulations can be run in the L3 cache of CPUs, on GPUs or FGPAs: hardware speed-up factor of s
- Parallelisation potential factor:

$$p = s\,\frac{N_{\rm tiles}}{r} = s\left(\frac{L}{L_{\rm sCOLA}}\right)^3$$

$$L = 1\ {\rm Gpc}/h$$
$$L_{\rm tile} = 125\ {\rm Mpc}/h$$
$$L_{\rm buffer} = 29.3\ {\rm Mpc}/h$$
$$r \approx 3.17$$
$$p \approx 161.59$$



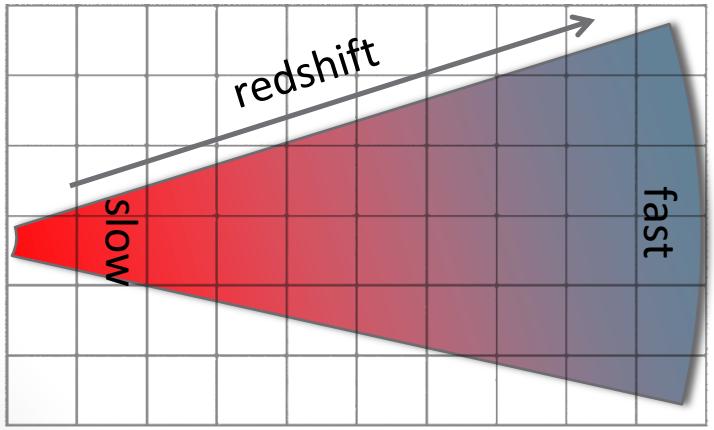




FL, Faure, Lavaux, Wandelt, Jaffe, Heavens, Percival & Noûs, 2003.04925

Additional benefits

- Light-cones and mock catalogues:
  - sCOLA boxes only need to run until they intersect the observer's past lightcone.
  - Most of the high-z volume will run faster than $z = 0$.
  - Many unobserved sCOLA boxes do not even have to run!
  - The wall-clock time limit is the time for running a single sCOLA box to $z = 0$ at the observer's position



- Gravity and physics models: any gravity model (e.g. P^3M, tree, or AMR) and non-gravitational physics (hydrodynamics) can be used within tiles
- Grid computing: the algorithm is suitable for inexpensive, strongly asynchronous networks
- Robustness to node failure

Conclusions

- In the age of peta-/exa-scale computing, we introduced a perfectly parallel and easily applicable algorithm for cosmological simulations using sCOLA, a hybrid analytical/numerical technique.
- The approach is based on a tiling of the full simulation box, where each tile is run independently.
- Resulting larger and higher-resolution cosmological simulations can be used in the context of Euclid and upcoming extremely large-scale surveys.
- The algorithm can benefit from a variety of hardware architectures. It is suitable for participatory computing platforms such as Cosmology@Home (with notable visibility/outreach benefits).

  https://www.cosmologyathome.org
- The algorithm is implemented in the Simbelmynë code. The development branch will be made publicly available at

  http://simbelmyne.florent-leclercq.eu - https://bitbucket.org/florent-leclercq/simbelmyne/