A&A 570, A20 (2014)
DOI: 10.1051/0004-6361/201423831

Astronomy & Astrophysics

A connected component-based method for efficiently integrating multi-scale *N*-body systems

Jürgen Jänes[1,⋆], Inti Pelupessy[2], and Simon Portegies Zwart[2]

[1] Faculty of Science, University of Amsterdam, PO Box 94216, 1090 GE, Amsterdam, The Netherlands
[2] Leiden Observatory, Leiden University, PO Box 9513, 2300 RA, Leiden, The Netherlands
e-mail: pelupes@strw.leidenuniv.nl

Received 18 March 2014 / Accepted 20 July 2014

ABSTRACT

We present a novel method for efficient direct integration of gravitational *N*-body systems with a large variation in characteristic time scales. The method is based on a recursive and adaptive partitioning of the system based on the connected components of the graph generated by the particle distribution combined with an interaction-specific time step criterion. It uses an explicit and approximately time-symmetric time step criterion, and conserves linear and angular momentum to machine precision. In numerical tests on astrophysically relevant setups, the method compares favourably to both alternative Hamiltonian-splitting integrators as well as recently developed block time step-based GPU-accelerated Hermite codes. Our reference implementation is incorporated in the HUAYNO code, which is freely available as a part of the AMUSE framework.

Key words. methods: numerical – stars: kinematics and dynamics – gravitation

1. Introduction

Direct integration of the classical *N*-body problem is an important tool for studying astrophysical systems. Examples include planetary systems, open and globular clusters dynamics, large-scale dynamics of galaxies, and structure formation in the universe. In many cases the calculations involve systems where the intensity of gravitational interactions spans multiple orders of magnitude with corresponding time scale variations. For example, the initial stages of cluster formation are now thought to resemble multi-scale fractal structures (Goodwin & Whitworth 2004), and stellar systems are invariably formed with a high fraction of binaries and hierarchical multiples that affect the dynamical evolution in crucial ways (Portegies Zwart et al. 2010).

In practice, integrating multi-scale systems requires specialised methods that vary the resolution at which we treat different parts of the simulation. The aim of this is to obtain a solution with an acceptable accuracy without unnecessarily spending computational resources on the slowly evolving parts of the simulation. In generic *N*-body integrators, this idea is most commonly implemented via *particle-based block time steps* – every particle in the system maintains an individual time step limited to discrete values in a power of two hierarchy. These block time steps are then typically used to determine the frequency of calculating the total force acting on a particle (e.g. McMillan 1986; Makino 1991; Konstantinidis & Kokkotas 2010).

While considerably speeding up calculations, particle-based block time steps are nevertheless limited in their ability to treat the extreme scale differences often present in *N*-body systems. Hence, complementary strategies, such as binary regularisation and neighbourhood schemes, have been devised. These approaches complicate the implementation of *N*-body integrators, and often introduce new method-specific free parameters. It is also unclear whether these combinations of multiple strategies represent the best possible approach for integrating multi-scale *N*-body systems. These issues provide a clear incentive to explore alternative methods.

In Pelupessy et al. (2012), we derived generic *N*-body integrators that recursively and adaptively split the Hamiltonian of the system. These methods show improved conservation of the integrals of motion by always evaluating partial forces between particles in different time-step bins symmetrically, and by using an approximately time-symmetric time-step criterion.

In the present work, we introduce a new Hamiltonian-splitting integration method that is particularly adept at integrating initial conditions with significant hierarchical substructure. Our approach is based on assigning time steps to individual interactions, followed by partitioning the system Hamiltonian based on a graph formed by the set of interactions that are faster than a fixed threshold time step. The successive partitioning produces closed Hamiltonians such that we can easily use specialised solvers for situations where more efficient solvers are available. Numerical experiments show that our integrator compares favourably to existing splitting methods even for an ordinary Plummer sphere where the prevalence of isolated subsystems is not immediately obvious. For astrophysically realistic systems explicitly chosen for their multi-scale substructure, the performance gains increase can be orders of magnitude. An implementation of the method is incorporated in the HUAYNO code, which is freely available as a part of the AMUSE framework (Portegies Zwart et al. 2013; Pelupessy et al. 2013) and which was used for the tests presented in this paper.

Our method is similar in spirit, and accelerates the calculation of the *N*-body problem for much the same reasons as the well known neighbour schemes. The main idea is to divide the total force acting on a particle into a fast and a slow

⋆ *Current address*: the Gurdon Institute and Department of Genetics, University of Cambridge, Cambridge CB3 0DH, UK

component based the distance to the given particle. Different approaches have been used for treating fast and slow components. The Ahmad-Cohen neighbourhood scheme (Ahmad & Cohen 1973) treats fast components with a more strict time step criteria. Alternatively, the PPPT scheme (Oshino et al. 2011) integrates fast components with a fourth-order Hermite method while using a leapfrog-based tree code for the long range interactions. The criteria for determining neighbourhood memberships are heuristics known to work in numerical experiments, e.g. a sphere with a fixed radius centred on the acting particle. These methods need to continuously update neighbourhood memberships as the system state changes throughout the simulation. In addition, neighbourhood schemes only make a single distinction between treating small subsystems such as hard binaries or many-body close encounters, and the large scale dynamics. It is difficult to generalise a neighbourhood scheme beyond a binary differentiation of the particle distribution.

In Sect. 2 we describe a bottleneck in existing general N-body splitting methods, and derive our novel splitting scheme that overcomes this bottleneck. Section 3 presents the results of numerical tests comparing of our method to existing approaches. Finally, in Sect. 4 we discuss possible improvements and extensions to our work, including the feasibility of integrating general N-body systems using purely interaction-specific time steps.

2. Method

2.1. Deriving time stepping schemes via Hamiltonian splitting

The Hamiltonian for a system of N particles $i = 1 \ldots N$ under gravitational interaction can be represented as a sum of momentum terms T_i and potential terms V_{ij}:

$$H(\boldsymbol{p}_i, \boldsymbol{q}_i) = T + V = \sum_{i=1}^{N} T_i + \sum_{\substack{i,j=1 \\ i<j}}^{N} V_{ij} \tag{1}$$

$$T_i = \frac{\left|\boldsymbol{p}_i\right|^2}{2m_i} \tag{2}$$

$$V_{ij} = -G\frac{m_i m_j}{\sqrt{q_{ij}^2 + \varepsilon^2}} \tag{3}$$

where m_i is the mass, $\boldsymbol{q}_i$ is the position and $\boldsymbol{p}_i$ is the momentum of the ith particle of the system, and $q_{ij} = \|\boldsymbol{q}_i - \boldsymbol{q}_j\|$. The evolution of the state of the system for a time step h is given formally by the flow operator $E_H(h) = \exp(h\mathbb{H})$ where $\mathbb{H}$ is the Hamiltonian vector field corresponding to H.

If the Hamiltonian H of the system is representable as a sum of two sub-Hamiltonians, $H = A + B$, we can approximate the time evolution under H with a sequence of time evolution steps under the sub-Hamiltonians A and B. A straightforward successive application of the time evolution under A followed by the time evolution under B gives a first-order approximation of the full time evolution under $A+B$, while a second-order accurate approximation can be obtained with one additional operator evaluation (Sanz-Serna & Calvo 1994, Sect. 12.4; also Hairer et al. 2006).

$$E_{A+B}(h) = E_A(h/2)E_B(h)E_A(h/2) + O\left(h^2\right). \tag{4}$$

The sub-Hamiltonian A is evolved in two steps of $h/2$ and the sub-Hamiltonian B is evolved in a single step h. We can take advantage of this property of the splitting formula by dividing terms associated with fast interactions into A and terms associated with slow interactions into B. We can proceed by applying this splitting procedure to different sub-Hamiltonians multiple times, thereby constructing an integrator that evaluates parts of the Hamiltonian at h, $h/2$, $h/4$ etc., similarly to the power of two hierarchy used in block time step schemes. This approach was followed in Pelupessy et al. (2012), below we introduce some notation and give a rough derivation of the integrators there.

Hamiltonians consisting of a single momentum term and Hamiltonians consisting of a single potential term have analytic solutions. For a momentum term of the ith particle

$$H_i(\boldsymbol{p}_i, \boldsymbol{q}_i) = T_i = \frac{\langle \boldsymbol{p}_i, \boldsymbol{p}_i \rangle}{2m_i} \tag{5}$$

the solution consists of updating the position of the ith particle under the assumption of constant velocity for a time period of h (all positions except the position of the ith particle and the momenta of all particles remain unchanged).

$$\boldsymbol{q}_i(t+h) = \boldsymbol{q}_i(t) + h\boldsymbol{v}_i(t). \tag{6}$$

We call the time evolution operator for the momentum term of the ith particle the *drift operator* and write D_{h,T_i}.

For a single potential term between particles i and j

$$H_{ij}(\boldsymbol{p}_i, \boldsymbol{q}_i) = V_{ij} = -G\frac{m_i m_j}{\sqrt{q_{ij}^2 + \varepsilon^2}} \tag{7}$$

the solution consists of updating the momenta of the ith and jth particles under the assumption of constant force for a time period of h (all momenta except the momenta of the ith and jth particles and the positions of all particles remain unchanged).

$$\boldsymbol{p}_i(t+h) = \boldsymbol{p}_i(t) + h\boldsymbol{F}_{ij}(t) \tag{8}$$

$$\boldsymbol{p}_j(t+h) = \boldsymbol{p}_j(t) + h\boldsymbol{F}_{ji}(t). \tag{9}$$

We call the time evolution operator for the potential term between the ith and jth particles the kick operator and write $K_{h,V_{ij}}$.

In addition to the kick and drift operators, the two-body Hamiltonian

$$H_{ij}(\boldsymbol{p}_i, \boldsymbol{q}_i) = T_i + T_j + V_{ij} = \frac{\langle \boldsymbol{p}_i, \boldsymbol{p}_i \rangle}{2m_i} + \frac{\langle \boldsymbol{p}_j, \boldsymbol{p}_j \rangle}{2m_j} - G\frac{m_i m_j}{\sqrt{q_{ij}^2 + \varepsilon^2}} \tag{10}$$

is solved (semi-) analytically by the Kepler solution[1].

In Pelupessy et al. (2012), we derive multiple integrators that recursively and adaptively split the system Hamiltonian through the second-order splitting formula (4). At every step in the recursion, all particles under consideration are divided into a slow set S and a fast set F by comparing the particle-specific time step function $\tau(i)$ to a pivot time step h.

$$S = \{i \in 1 \ldots N : \tau(i) \geq h\} \tag{11}$$

$$F = \{i \in 1 \ldots N : \tau(i) < h\}. \tag{12}$$

Using the two sets S and F, we can rewrite the system Hamiltonian as follows.

$$H = H_S + H_F + V_{SF}. \tag{13}$$

The sub-Hamiltonian H_S can be thought of as a "closed Hamiltonian" of the particles in S. Specifically, it consists of all

[1] Even the case with $\varepsilon \neq 0$ can be solved in a universal variable formulation (Ferrari, priv. comm.).

drifts of particles in S and all kicks where both participating particles are in S. The same property holds for the sub-Hamiltonian H_F and the particles in F. The mixed term V_{SF} contains all kicks where one particle is in S and the other is in F.

We proceed by applying the second-order splitting rule (4):

$$E_{h,H} = E_{h,H_S+H_F+V_{SF}} \tag{14}$$
$$\approx E_{h/2,H_F} E_{h,H_S+V_{SF}} E_{h/2,H_F} \tag{15}$$

(this is not the only conceivable approximation). The sub-Hamiltonian H_F is closed, and consists of particles where $\tau(i) < h$. We integrate H_F by recursively applying the entire "slow/fast" partitioning, but using a smaller pivot $h/2$. In contrast, both $H_S = T_S + V_S$ and V_{SF} are explicitly decomposed into individual kicks and drifts which are applied using the current pivot time step h. We refer to this particular choice as the HOLD method (since it "holds" V_{SF} for evaluation at the slow timestep).

$$E_{h,H_S+V_{SF}} = E_{h,T_S+V_S+V_{SF}} \tag{16}$$
$$\approx D_{h/2,T_S} K_{h,V_S+V_{SF}} D_{h/2,T_S}. \tag{17}$$

The pivot time step h is halved with each consecutive partitioning, and the recursion terminates when all remaining particles are placed into the S set.

As noted previously, recursively and adaptively splitting the system Hamiltonian using the second order splitting rule (Eq. (4)) is similar to conventional block time steps. Both approaches evolve different parts of the system using time steps that belong to a power of two hierarchy. However, the Hamiltonian splitting method derived above evaluates pairwise particle forces symmetrically in the sense that a "kick" from particle i to particle j (Eq. (8)) is always paired with an opposite kick from particle j to particle i (Eq. (9)). Furthermore, the kicks acting upon a particle at any given timestep typically correspond to partial forces only. This is in contrast to conventional block time steps where we always calculate the *total* force acting on a particle at the frequency determined by the particle-specific time step criteria $\tau(i)$

$$\boldsymbol{p}_i(t+h) = \boldsymbol{p}_i(t) + h \sum_{j=1, j\neq i}^{N} \boldsymbol{F}^{\star}_{ij}(t) \tag{18}$$

where, $\boldsymbol{F}^{\star}_{ij}(t)$ is the force acting on particle i due to particle j, derived from extrapolated positions if necessary. Specifically, in situations where the position of particle j has not been calculated for time t, we calculate the force by extrapolating the position at t from the last known position. This can happen when particle i is assigned a smaller time step than particle j. We refer to this method as BLOCK, and include it as a reference in our numerical tests to determine whether more "aggressive" splitting methods (such as HOLD) reduce the number of kicks and drifts while maintaining the accuracy of the solution.

The HOLD method evolves all kicks between fast particles at the fast time step. This is inefficient in the presence of isolated fast subsystems, as interactions between particles that belong to different subsystems could be evolved at a slower time step. As an extreme example, consider a Plummer sphere with each star being replaced by a stable hard binary. Here, every star has a close binary interaction that needs to be evaluated at a fast time step. However, the HOLD integrator will in this case integrate all interactions, including long-range interactions between stars in different binaries at a time step determined the binary interactions. The behaviour of the method becomes equivalent to evolving the entire system with a shared global time step!

In addition to the dramatic example just discussed, the same inefficiency – evaluating long-range interactions between isolated fast subsystems at time steps determined by fast interactions inside the subsystems – can manifest itself in other situations, such as the following.

- In a system with multiple globular clusters, each individual globular cluster is a subsystem.
- In a globular cluster with planets around some of the stars, each star with planets is a subsystem.
- In a single globular cluster, each close encounter between two or more stars is a subsystem.

2.2. Hamiltonian splitting with connected components

The partitioning used in the HOLD method is based on a *particle-specific* time step criteria $\tau(i)$, which by definition cannot separate slow and fast interactions in situations where all particle-specific time steps have the same (fast) value. We therefore introduce the *interaction-specific* time step criterion $\tau(i, j)$

$$\tau(i,j) = \eta \min\left(\frac{\tau_{\text{freefall}}(i,j)}{\left(1 - \frac{1}{2}\frac{d\tau_{\text{freefall}}(i,j)}{dt}\right)}, \frac{\tau_{\text{flyby}}(i,j)}{\left(1 - \frac{1}{2}\frac{d\tau_{\text{flyby}}(i,j)}{dt}\right)} \right) \tag{19}$$

where $\tau_{\text{freefall}}(i, j)$ and $\tau_{\text{flyby}}(i, j)$ are proportional to the interparticle free-fall and interparticle flyby times as defined by Eqs. (13) and (16) in Pelupessy et al. (2012), and η is an accuracy parameter.

We split the system Hamiltonian using the connected components (Cormen et al. 2001, Sect. B.4) of the undirected graph generated by the time step criteria $\tau(i, j)$. Specifically, the particles of the system correspond to the vertices of the graph, and there is a edge between particles i and j if their interaction cannot be evaluated at the threshold time step h.

$$\tau(i,j) < h. \tag{20}$$

Figure 1 we visualise the time step graphs at varying values of the pivot time step h for three different fractal initial conditions with different fractal dimension (described further in Sect. 3). As the pivot time step h decreases, the set of interactions (and associated particles) that cannot be evaluated at the current pivot time step gradually decreases as well. Although for visualisation purposes, we plot the time step graph of the entire system for varying h, the connected components (CC) splitting method we are about to introduce typically calculates CC for the entire system only once, at the largest pivot time step. At smaller pivot time steps, the connected components search is only calculated for parts of the system. The intuition behind this partitioning comes from clustering by maximising the margin between individual clusters as described in Duan et al. (2009).

Given a fixed pivot time step h, let the sets C_i, $i = 1 \ldots K$ contain vertices of K non-trivial connected components, and the set R ("remainder set") contain all particles in trivial connected components[2]. Based on the particle sets C_i and R, we rewrite the Hamiltonian of the system in the following form

$$H = H_C + H_R + V_{\text{CC}} + V_{\text{CR}} \tag{21}$$

[2] A trivial connected component is a connected component with exactly one vertex and a non-trivial connected component is a connected component with at least two vertices.

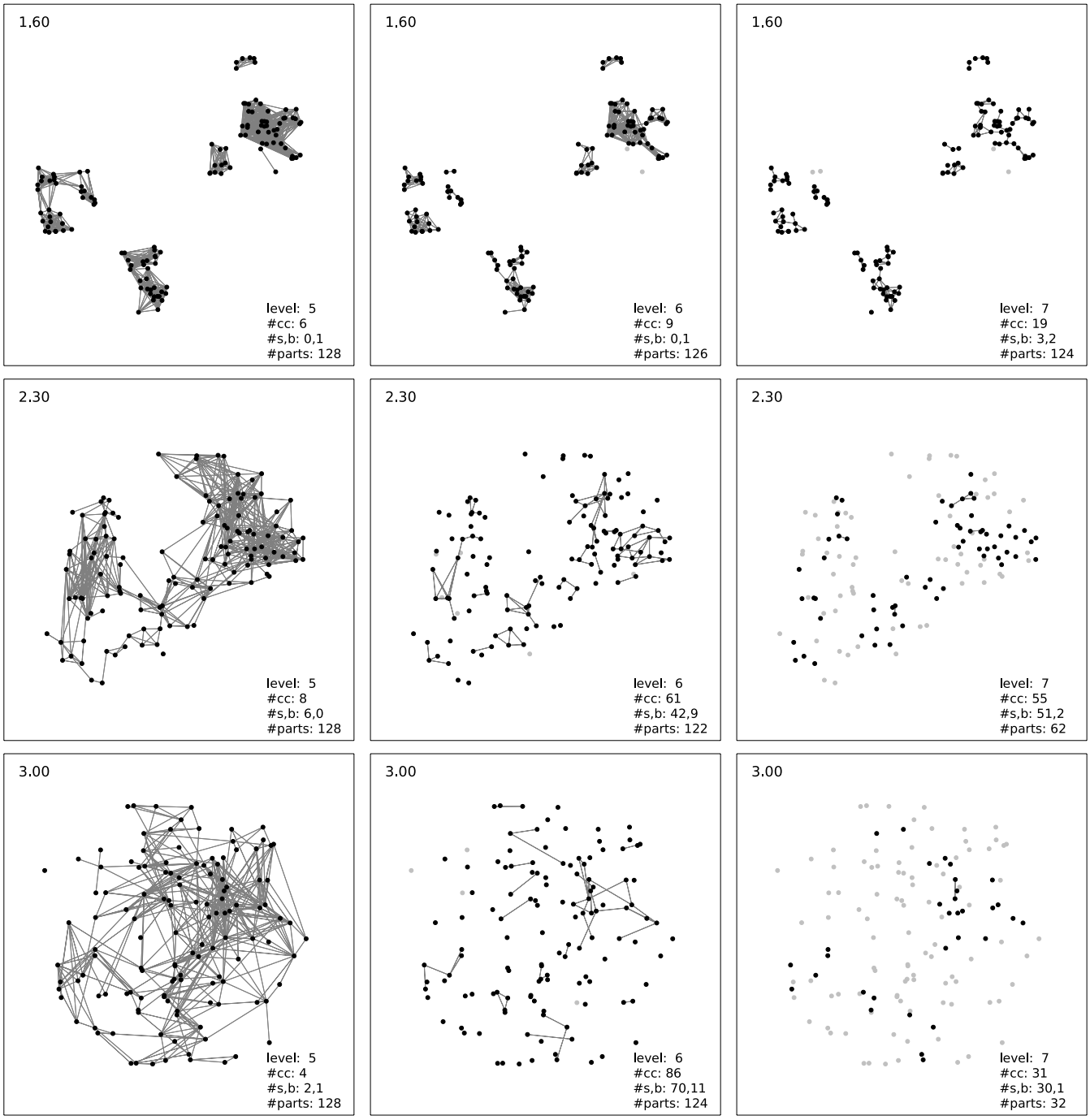

Fig. 1. Time step graphs generated by $\tau(i, j)$ at different levels of the time step hierarchy (*left to right*) for three values (*top to bottom rows*) of the fractal dimension. We plot particles that have been passed on as a part of a connected component from the previous time step level as black, grey points indicate points that are inactive on a given level (because they formed a single or binary component at a lower level). Thin grey lines indicate interactions with $\tau(i, j) < h$. Indicated in each frame are the fractal dimension (*top left*), and (on the *bottom right*) the level in the hierarchy, the number of connected components (cc) at this level, as well as the number of components that are single (s) and binary (b), and the total number of particles.

where the individual terms are defined as follows.

$$H_C = \sum_{i=1}^{K} H_{C_i} = \sum_{i=1}^{K} \left(\sum_{j \in C_i} T_j + \sum_{\substack{j,k \in C_i \\ i<j}} V_{jk} \right) \tag{22}$$

$$H_R = T_R + V_R = \sum_{i \in R} T_i + \sum_{\substack{i,j \in R \\ i<j}} V_{ij} \tag{23}$$

$$V_{\mathrm{CC}} = \sum_{\substack{i,j=1 \\ i<j}}^{K} V_{C_i C_j} = \sum_{\substack{i,j=1 \\ i<j}}^{K} \left(\sum_{\substack{k \in C_i \\ l \in C_j}} V_{kl} \right) \tag{24}$$

$$V_{\mathrm{CR}} = \sum_{i=1}^{K} V_{C_i R} = \sum_{i=1}^{K} \left(\sum_{\substack{j \in C_i \\ k \in R}} V_{jk} \right). \tag{25}$$

```
evolve_cc(H, h):
    // split_cc() decomposes particles in H (eq 25) into:
    // 1) K non-trivial connected components C_1..C_K
    // 2) Rest set R
    (C_1..C_K, R) = split_cc(H, h)

    // Independently integrate every C_i at reduced pivot time step h/2 (eq 27)
    for C_i in C_1..C_K:
        evolve_cc(C_i, h/2)

    // Apply drifts and kicks at current pivot time step h (eq 30)
    drift(R, h/2) // evolves T_R
    kick(R, R, h) // evolves V_RR
    kick(C_1..C_K, C_1..C_K, h) // evolves V_CC (eq 23)
    kick(C_1..C_K, R, h) // evolves V_CR (eq 24)
    drift(R, h/2) // evolves T_R

    // Independently integrate every C_i at reduced pivot time step h/2 (eq 27)
    for C_i in C_1..C_K:
        evolve_cc(C_i, h/2)
```

Fig. 2. Pseudocode for a second-order CC splitting routine for integrating a set of particles `H` for time step `h`.

The term H_C is the sum of all closed Hamiltonians H_{C_i}, each corresponding to one of the K connected components. In every H_{C_i} all drifts and some kicks cannot be evolved at the time step h without violating the time step criteria. The term H_R consists of the closed Hamiltonian formed by all of the particles in the rest system. All drifts and kicks in H_R can be evolved at the current time step h.

The term $V_{\rm CC}$ contains all kicks between particles that are in *different* connected components. These kicks can be evaluated at the time step h. We point out that $V_{\rm CC}$ explicitly contains the terms that are evolved inefficiently in the HOLD method. Similarly, $V_{\rm CR}$ contains all kicks where one of the particles is in a connected component C_i, and the other is in the rest set R.

We split the system Hamiltonian H by applying the second-order splitting rule (4):

$$E_{h,H} = E_{h,H_C+H_R+V_{\rm CC}+V_{\rm CR}} \tag{26}$$
$$\approx E_{h/2,H_C}\, E_{h,H_R+V_{\rm CC}+V_{\rm CR}} E_{h/2,H_C} \tag{27}$$

such that individual connected components C_i are independently evolved at a higher pivot time step $h/2$ via recursion.

$$E_{h/2,H_C} = \prod_{i=1}^{K} E_{h/2,H_{C_i}}. \tag{28}$$

All remaining terms (including $V_{\rm CC}$) are decomposed into individual drifts and kicks using the second-order splitting rule (4).

$$E_{h,H_R+V_{\rm CC}+V_{\rm CR}} = E_{h,T_R+V_R+V_{\rm CC}+V_{\rm CR}} \tag{29}$$
$$\approx D_{h/2,T_R} K_{h,V_R+V_{\rm CC}+V_{\rm CR}} D_{h/2,T_R} \tag{30}$$
$$= D_{h/2,T_R} K_{h,V_R} K_{h,V_{\rm CC}} K_{h,V_{\rm CR}} D_{h/2,T_R}. \tag{31}$$

As with the HOLD method, the pivot time step h is halved at each successive partitioning such that at some point, all remaining particles are in the remainder set R.

Finally, the partitioning can lead to situations where H_{C_i} is a two-body Hamiltonian (Eq. (10)). We can use this property by integrating these cases with a dedicated a Kepler solver (we discuss this further in Sect. 3.2).

2.3. Implementation

We implemented the CC split in the HUAYNO code, which is freely available as a part of the AMUSE framework. Figure 2 sketches the main routine of the CC integrator in pseudocode, including explicit references to the corresponding equations and variables used in the derivation of the method (Sect. 2.2).

Subroutines and data structures that store the system state, calculate time steps, apply kicks and drifts to groups of particles, and gather statistics, are shared with other integrators such as the HOLD method. All particle states are kept in a contiguous block of memory. The connected component algorithm is implemented as a breadth-first search. It reshuffles particle states such that particles in the same connected component or rest set are kept adjacent to each other. Connected components are represented by a start and an end pointer to the contiguous array of particle states.

The time complexity of the connected component decomposition for N particles has an upper bound of $O(N^2)$. This matches the time complexity of the splitting step of the HOLD method – while the actual shuffling of the particles into S and F sets is $O(N)$, this division is based on the preceding step of calculating particle-based time steps $\tau(i)$ for all particles, which is $O(N^2)$.

For the special case where all interactions between the N particles are below the threshold h, the complexity of the connected components decomposition is $O(N)$. This can happen multiple times (at consecutive recursion levels) when the initial value of the pivot time step h is sufficiently large. Figuratively, if particle X has a known connected component while particle Y is unassigned, we can assign particle Y to the connected component of particle X based on a single time step evaluation $\tau(X, Y)$. A key step of the connected components search is choosing a particle X with a known connected component, followed by assigning the membership of X to all unassigned particles U_k where $\tau(X, U_k) < h$. For the special case under consideration, a single iteration of this step is sufficient to assign membership to all particles (irrespective of the choice of the initial particle X), leading to a time complexity of $O(N)$. Further, while the splitting step is bounded from above by $O(N^2)$ for both HOLD and CC, the HOLD split always calculates time steps for all interactions. This is not the case with the CC method, and numerical tests in

Table 1. Overview of the integrators used in the numerical tests.

Method	Description
BLOCK	Conventional particle-based block time steps – positions of particles in lower time step bins are extrapolated when calculating the movement of particles in faster time step bins.
HOLD	Individual timestepping method based on Hamiltonian splitting. Particles in different timestep bins interact by exchanging symmetric kicks (Pelupessy et al. 2012).
CC	An implementation of the connected components splitting (Sect. 2.2). Iterative partitioning based on the connected components of the graph generated by the pairwise timestep criterion.
CC_KEPLER	An extension of the CC method that uses a Kepler solver to evolve connected components with two particles (Sect. 3.2).

Notes. The implementations share a significant amount of the code and data structures used for storing system state, calculating time steps, and evaluating (partial) forces.

Sect. 3 indicate that the reduction in time step evaluations does translate into improved performance.

3. Tests

We present results of numerical experiments of the CC splitting method described in the previous section. We confirm that the CC method works as intended conceptually by comparing to alternative Hamiltonian splitting methods (see Table 1 for an overview). Specifically, we demonstrate that the connected components search does not use excessive computational resources, reduces the number of elementary operations (kick, drift and time step evaluations) while maintaining the accuracy of the solution, and performs particularly well on multi-scale problems. Finally, we compare the CC method to established N-body codes. We use N-body units as described in Heggie & Mathieu (1986).

3.1. Smoothed Plummer sphere test

We begin by integrating an equal-mass 1024-body Plummer sphere with softening ($\varepsilon = 1/256$) for 700 N-body time units using a time step accuracy parameter of $\eta = 0.01$. We choose initial velocities such that the Plummer sphere is in a dynamic equilibrium. This setup is chosen to match the long-term integration tests in Nitadori & Makino (2008, their Sect. 3.2).

Figure 3 visualises the conservation of the integrals of motion, the time evolution of the mass distribution, and performance metrics. While all three methods show similar energy conservation properties, only HOLD and CC maintain centre of mass, linear momentum and angular momentum near machine precision. As noted previously in Pelupessy et al. (2012), this is caused by unsynchronised kicks which are only present in the BLOCK scheme. The solutions obtained by all three methods reproduce known results in terms of Lagrangian radii, the core radius and the core density. The CC scheme is about twice as fast than the HOLD scheme at the beginning of the simulation, and remains the fastest scheme throughout the run. The overall runtime measurements correlate with the number of time step formula evaluations and, to a lesser extent, the number of kick and drift formula evaluations. This indicates that the improved runtime is attributable to a reduction of time step, kick and drift formula evaluations.

The left plot of Fig. 4 visualises energy error of evolving the softened Plummer sphere as described previously, but for 1 N-body units and under varying time step accuracy η. As predicted, all three methods show second order behaviour. On the corresponding wall-clock time vs energy error plot on the right CC consistently outperforms HOLD, followed by BLOCK. We emphasise that the Plummer sphere is a spherically symmetric configuration with a smoothly changing mass distribution, and a non-zero softening length ε sets an upper limit on the hardness of the binaries that can form during the simulation. Hence, we would not expect the CC scheme to have a significant advantage over the HOLD method.

3.2. Unsoftened Plummer sphere test

We proceed by evolving an equal-mass 1024-body Plummer sphere without softening through core collapse. We choose initial velocities consistent with a dynamic equilibrium, as in the softened case considered previously. This setup is chosen to match a test used on a modern implementation of a fourth-order Hermite scheme with block time steps in Konstantinidis & Kokkotas (2010, their Sect. 3.4.1).

In addition to the HOLD and CC schemes that have been introduced previously, we also test a modification of the CC scheme with a dedicated Kepler solver (CC_KEPLER). In this scheme a Kepler solver is used for evolving connected components consisting of two particles. This is a form of algorithmic regularisation of binaries, but note that the regularisation follows naturally from the structure of the integrator and no separate binary detection or additional free parameters are necessary. The implementation of the Kepler solver is based on a universal variable formulation (Bate et al. 1971).

Results of the core collapse simulation are visualised in Fig. 5. All three methods produce solutions that are realistic in terms of the evolution of the mass distribution. Energy conservation is comparable to what is observed in Konstantinidis & Kokkotas (2010). Other integrals of motion show conservation around machine precision with the exception of a jump in the HOLD method around core collapse (this is caused by a high speed particle escaping from the system, causing a loss of precision in the force evaluations).

Before core collapse, execution times are roughly equivalent to the softened case considered previously (Sect. 3.1) – CC shows a modest improvement over HOLD, and CC_KEPLER is very close to CC. Around core collapse, execution times of the HOLD and CC methods gradually increase by an order of magnitude (the CC method still consistently outperforms the HOLD method). In contrast, execution time used by the CC_KEPLER method remains relatively uniform throughout the simulation, including core collapse.

The Sakura integrator achieves a similarly efficient treatment of close binaries by decomposing the evolution of an N-body Hamiltonian into a sequence of Kepler problems

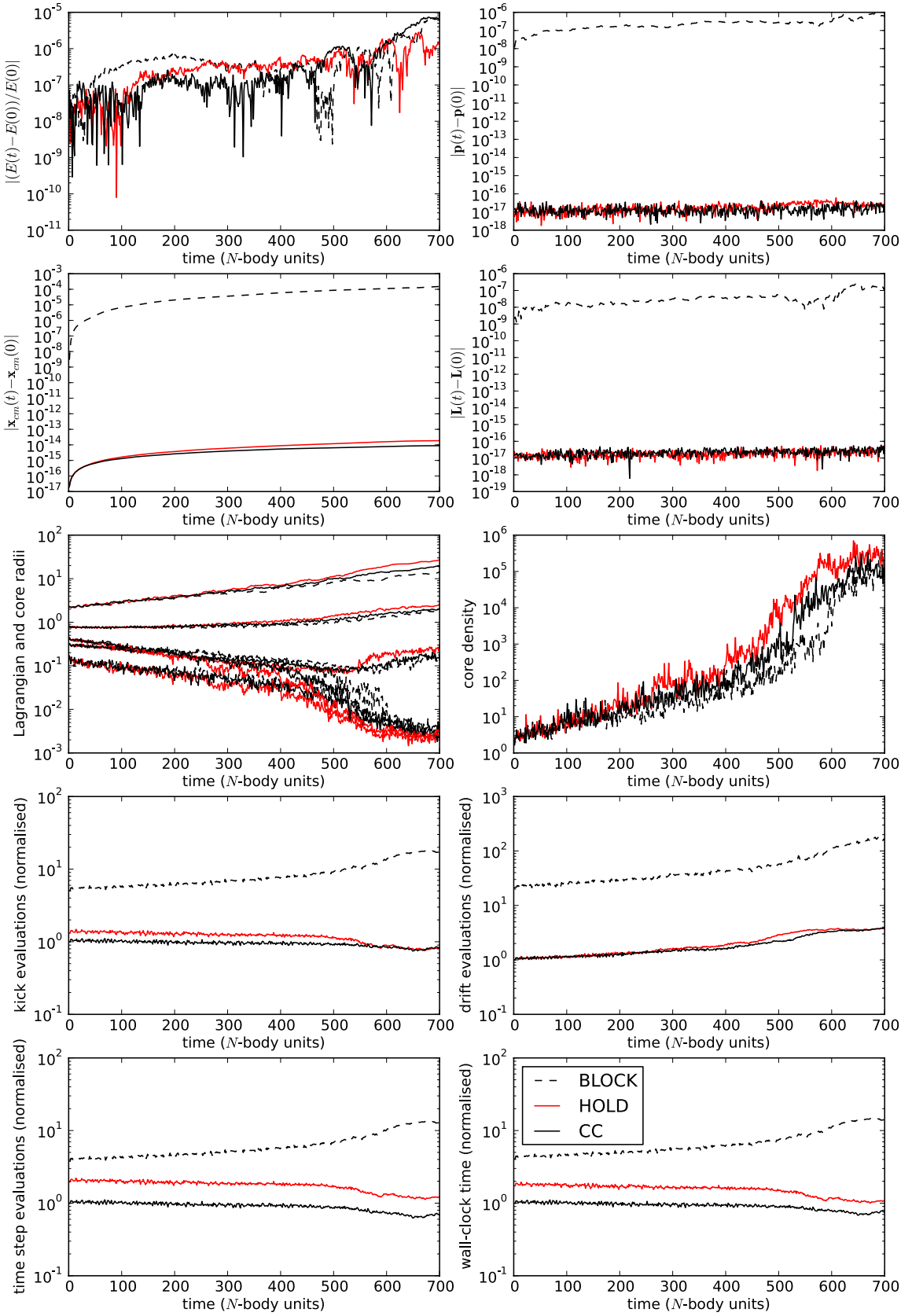

Fig. 3. A comparison of the BLOCK, HOLD and CC methods on integrating an softened 1024-body Plummer sphere: conservation of the integrals of motion (*top two rows*), evolution of the mass distribution of the solution (*middle row*) and performance metrics (*two bottom rows*). Lagrangian radii are plotted for 90%, 50%, 10% and 1% of the system mass. The performance metrics are normalised to the CC method at the start of the simulation. The CC method performs 1.1×10^9 kick, 6.6×10^8 time step, and 5.0×10^6 drift evaluations, taking 62 s for the first global time step (1/512-th of the simulation) on a laptop with a 1.3 GHz Intel Core i5 processor.

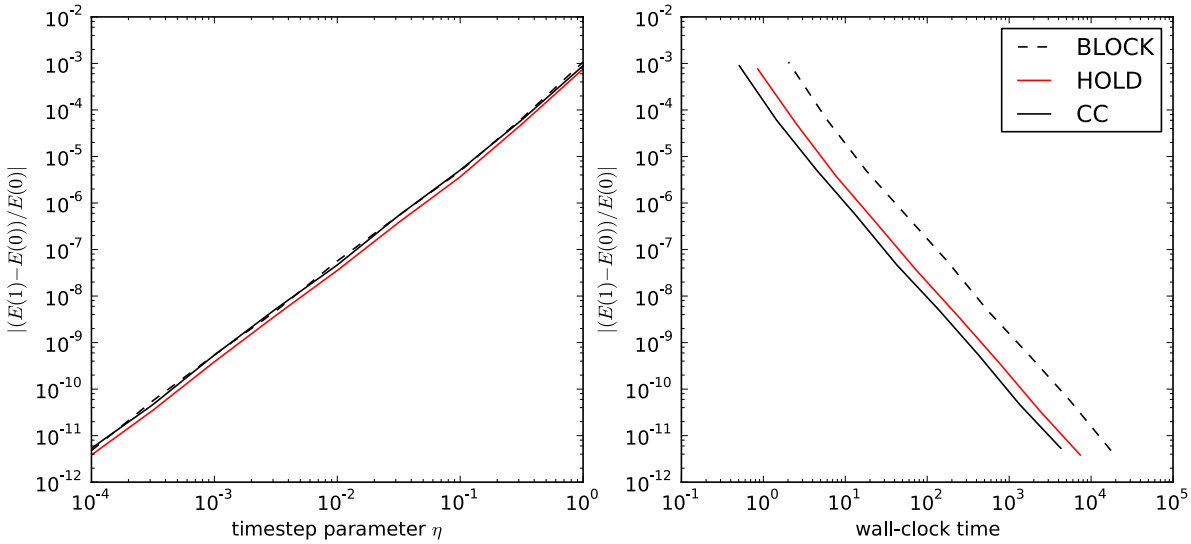

Fig. 4. *Left*: time step accuracy parameter η vs. energy error for integrating a 1024-body Plummer sphere for 1 N-body units. *Right*: corresponding wall-clock time vs. energy error from the same set of tests.

(Gonçalves Ferrari et al. 2014). The main source of errors in Sakura comes from many-body close encounters, as these are difficult to decompose into two-body interactions. In contrast, CC_KEPLER only uses the binary solver for an isolated binary system, and switches to the regular many-body integrator when necessary (this is further discussed in Sect. 4.1).

3.3. Fractal distributions

Since our new methods are based on the partitioning of the particle distribution in connected subsystems, we expect the method to be especially well suited to situations where substructure with extreme density contrasts exist. We therefore proceed by integrating a set of initial conditions developed with the aim of describing a star cluster with fractal substructure (Goodwin & Whitworth 2004). These initial conditions mimic the observed distribution of young stellar associations. They are parametrized by a fractal dimension: a low fractal dimension leads to an inhomogeneous ("structured") distribution of stars whereas a high fractal dimension leads to a more homogenous ("spherical") distribution (Fig. 1). For the highest possible fractal dimension value of 3, the initial conditions approximate a constant density sphere. We use $\eta = 0.03$, and integrate a 1024-particle system under an unsoftened potential for 0.25 N-body units for varying fractal dimensions.

In Fig. 6 we plot the energy error and runtime of the simulation, averaged over 10 runs, as a function of the fractal dimension f. While all integrators show similar energy conservation, CC and CC_KEPLER consistently outperform BLOCK and HOLD irrespective of fractal dimensions in terms of runtime. Further, runtime increases for decreasing fractal dimension for the BLOCK and HOLD integrators, while runtime remains essentially flat (and even decreases slightly) for decreasing fractal dimension for the CC and CC_KEPLER integrators.

3.4. Plummer sphere with binaries

We proceed by looking at how our methods perform on systems containing a large number of binaries. Specifically, we take a Plummer sphere and replace every particle with a binary system. The positions and velocities of the particles are chosen such that under the absence of external perturbations, they would form a stable binary with a randomly oriented orbital plane, and a semi-major axis drawn uniformly in log space between $log(a)$ and -0.5. We integrate a system of 512 binaries (=1024 individual particles) for 0.25 N-body units with $\eta = 0.03$.

Figure 7 visualises energy conservation and runtime of the initial conditions as a function of minimum semi-major axis a. For large a, the introduced binaries are generally unbounded, and the results are equivalent to evolving an ordinary Plummer sphere. As the minimum a decreases, the introduced binaries become bounded and their interactions start dominating in the integration time, leading to a significant advantage for CC and CC_KEPLER methods.

3.5. Cold collapse test

As a final test we evaluate the performance of our integrators in a cold collapse scenario. Specifically, we use the fractal initial conditions described in Sect. 3.3 with the initial velocities set to zero. We consider a "structured" case with the fractal dimension $f_d = 1.6$, and a "spherical" case with $f_d = 3.0$. We evolve initial conditions for 2 N-body time units. For the spherical case, this is past the moment of collapse that occurs around 1.5 N-body time units. For the structured case the moment of collapse is less well-defined, as different substructures collapse at different times.

We compare CC and CC_KEPLER to two recent N-body codes, Ph4 (McMillan, in prep.) and HiGPUs (Capuzzo-Dolcetta et al. 2013). Both codes use a Hermite scheme with conventional block time steps. Ph4 implements a fourth-order scheme with the option of using the GPU-accelerated SAPORRO library (Gaburov et al. 2009; and Bédorf et al., in prep.). HiGPUs implements a sixth-order scheme and requires a GPU to run. We conduct our tests on a workstation – running on a single core of an Intel i7-2720QM CPU, and a GTX460M GPU. The FLOPS performance of the GPU is roughly 40 times larger than a single core of the CPU. The hardware setup is thus indicative only of the intrinsic algorithmic scaling, rather than representative of the performance in production simulations (which would use multiple and/or more powerful CPUs/GPUs). We use unsoftened potential for CC, CC_KEPLER, and Ph4 without

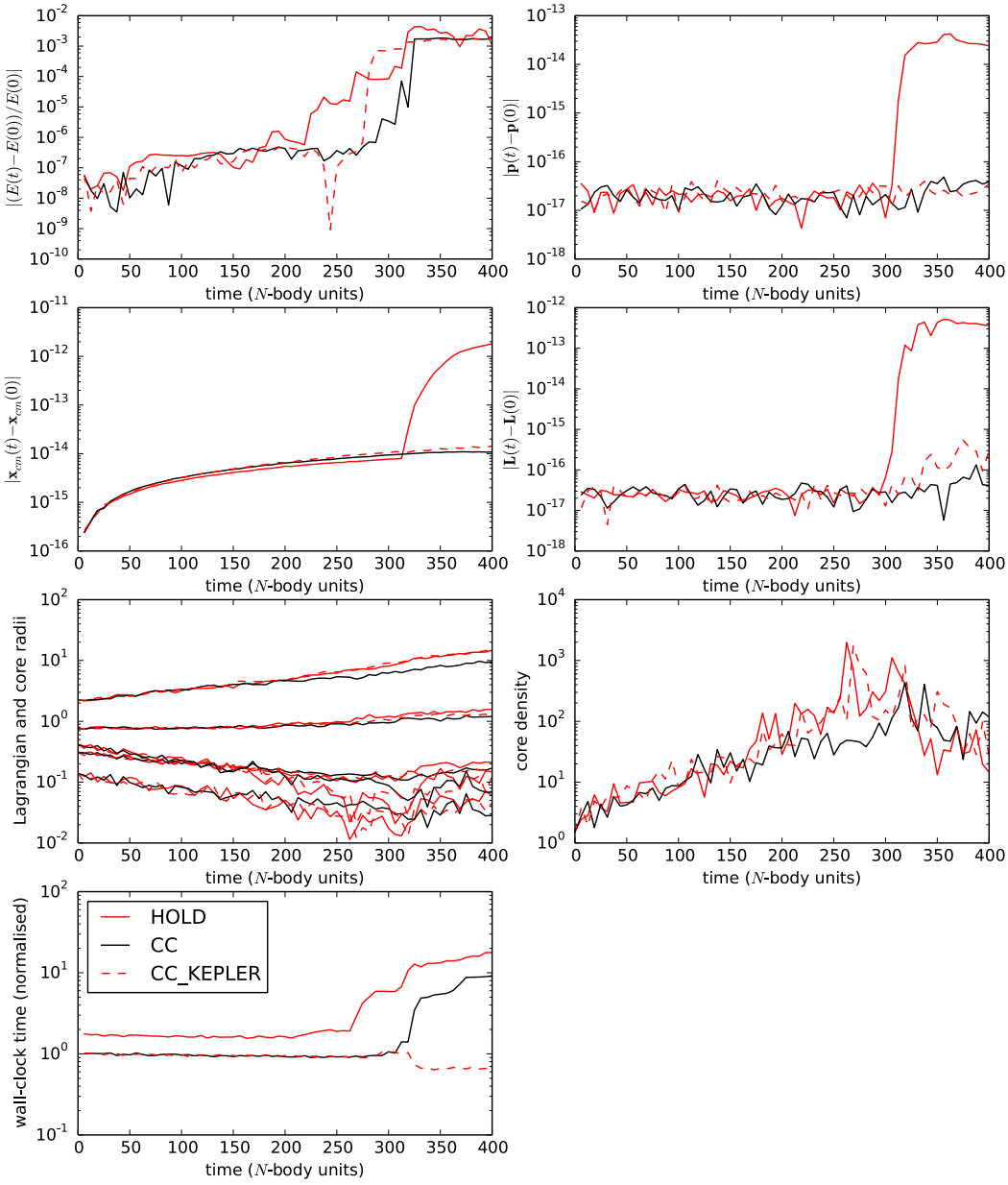

Fig. 5. A comparison of the HOLD, CC and CC_KEPLER methods on integrating an unsoftened 1024-body Plummer sphere through core collapse: conservation of the integrals of motion (*top two rows*), evolution of the mass distribution of the solution (*third row*) and wall-clock time (*bottom left*). Lagrangian radii are plotted for 90%, 50%, 10% and 1% of the system mass. Wall-clock times are normalised by CC_KEPLER at the first global time step (1/64-th of the simulation). On a laptop with a 1.3 GHz Intel Core i5 processor, CC_KEPLER integrates the first global time step in roughly 5 min while the entire simulation takes about 7 h.

GPU acceleration. We use a very small softening parameter ($\varepsilon = 10^{-4}$) for Ph4 with GPU acceleration, and HiGPUS, as these run into severe slowdowns and/or crashes with unsoftened gravity – probably because of the limited precision of their GPU kernels. We set code-specific time step accuracy parameters to $\eta = 0.01$ for CC/CC_KEPLER, $\eta_4 = 0.1$ for Ph4, and $\eta_4 = 0.05$ and $\eta_6 = 0.6$ for HiGPUs.

Figure 8 visualises energy conservation, momentum conservation and the wall-clock time of the initial conditions as a function of the system size for structured and homogenous initial conditions. In the spherical case, setups that take advantage of the GPU (PH4_gpu and HiGPUs) outperform the alternatives, but note that both CC and CC_KEPLER show very similar scaling to Ph4 without GPU acceleration. In contrast, for structured case, CC_KEPLER and CC show a marked speed up in comparison with the conventional block time step schemes, being faster for this particular calculation than Ph4_GPU and HiGPUs, despite the latter having the advantage of using the GPU acceleration and integrating with softened gravity. The differences between the structured and the spherical cases highlight the relative advantage that the connected component approach has with respect to conventional block time steps when applied to multi-scale initial conditions.

4. Discussion

4.1. Using and extending the CC method

We introduced a novel method for direct integration of N-body systems based on splitting the system Hamiltonian using

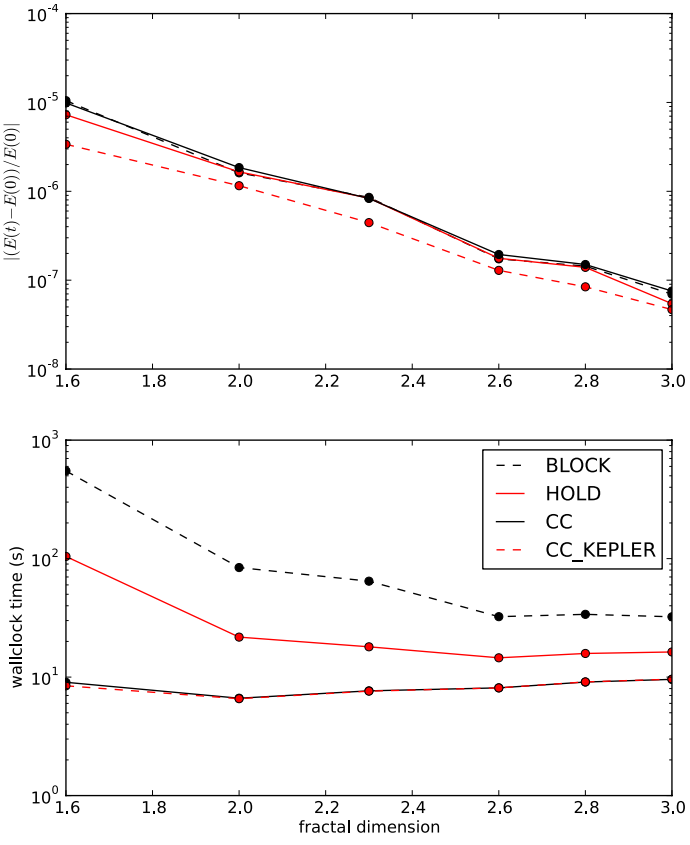

Fig. 6. Energy conservation (*top*) and runtime (*bottom*) for integrating a 1024-particle system of fractal initial conditions for 0.25 N-body units as a function of the fractal dimension.

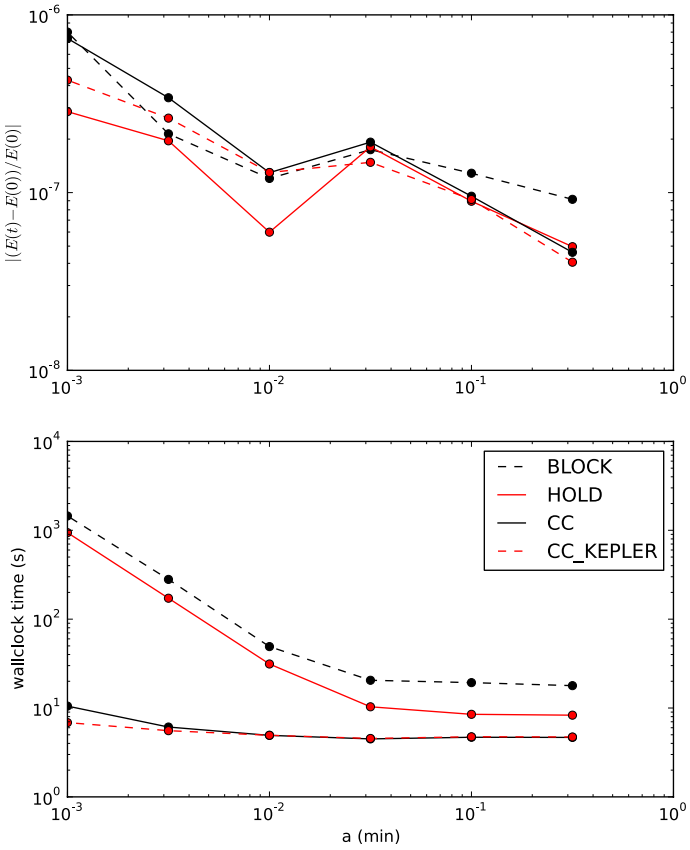

Fig. 7. Energy conservation (*top*) and runtime (*bottom*) for integrating 512-binary (=1024-particle) Plummer sphere for 0.25 N-body units as a function of the initial semi-major axis a.

connected components of the time step graph (the CC split). We were motivated by the need for a more efficient divide-and-conquer strategy for reducing the intractable Hamiltonian (Eq. (18)) to the least possible number of analytically solvable Hamiltonians (Eqs. (7), (6)). In comparison to existing splitting methods, notably the HOLD split introduced in Pelupessy et al. (2012), the CC split is particularly effective at splitting multi-scale systems. We have not encountered a situation where the HOLD split would be preferable over the CC split. The practical advantages of Hamiltonian splitting are similar to what is usually achieved with block time-steps. However, as our splitting methods, including the CC method, do not extrapolate particle states for evaluating the total force acting on a particle, we conserve linear and angular momentum to machine precision.

We went on to show on the example of the CC_KEPLER method that the connected components partitioning has additional uses beyond improved splitting efficiency. Specifically, we were able to incorporate regularisation of two-body close encounters by simply checking for the condition where the successive partitioning leads to a connected component with two particles, and evolving the corresponding two-body Hamiltonian (Eq. (10)) using a dedicated Kepler solver. This approach can be extended to many-body close encounters by using a suitable specialised solver (or e.g. chain regularisation methods, Mikkola 2008) to evolve isolated Hamiltonians corresponding to connected components with certain properties. Possible selection criteria include having a specific number of particles and/or a maximum time step below a threshold value.

The numerical experiments of Sect. 3 were chosen to mainly study the splitting aspect of N-body integration. We focused on normalised performance metrics, and the scaling of the wall-clock time as a function of the "multi-scaleness" in the initial conditions. Our current implementations would benefit from additional optimisations typically used in production-level N-body codes. Specifically, there is inherent parallelism in the CC method, as recursive calls for evolving successively smaller closed Hamiltonians only affect the state of the particles in the "current" closed component. It may be possible to parallelise the method based on this property. However, tests show that a naive approach does not scale well due to load-balancing issues, as subsystems can vary substantially in size. Alternatively, it could be feasible to implement the CC method on a GPU, as the major components – N-body force evaluation (Portegies Zwart et al. 2007; Belleman et al. 2008; Capuzzo-Dolcetta et al. 2013) and graph processing algorithms (Harish & Narayanan 2007) – have individually been successfully implemented on GPUs.

It may be possible to speed up the evaluation of long-range interactions between different connected components ($V_{\rm CC}$ in the CC decomposition formula) through a centre-of-mass (or multipole) approximation that form the basis of tree codes Barnes & Hut (1986). As long-range interactions between two connected components are evaluated symmetrically, this approach could make it possible to obtain most of the speedup of a tree code while maintaining good linear and angular momentum conservation. A potential pitfall with this approach could arise from the fact that the time step criterion $\tau(i, j)$ used in finding the connected components is only partially determined by the

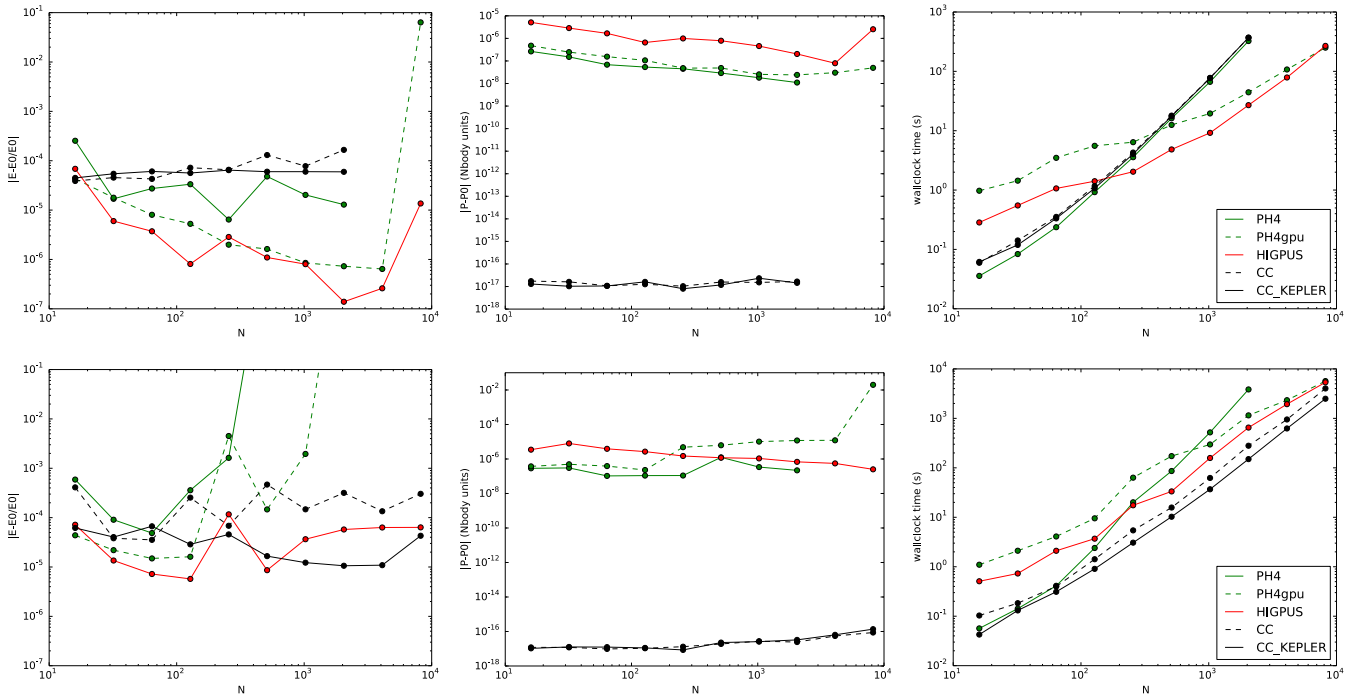

Fig. 8. Energy conservation (*left column*), momentum (*middle column*) and wall-clock time (*right column*) as a function of system size N for the cold collapse test. The *top row* shows results for a homogenous sphere (fractal dimension $f_d = 3$), while the *bottom row* shows a highly structured fractal ($f_d = 1.6$). We evolve the initial conditions for 2 N-body time units, and plot the mean values across 5 runs. CC, CC_KEPLER and Ph4 use unsoftened gravity ($\varepsilon = 0$), while PH4_gpu and HIGPUS use very small softening ($\varepsilon = 10^{-4}$).

coordinates of the particles. As such, particles in the same connected component may occupy a "non-compact" region in physical space, making multipole approximation difficult.

4.2. Formally optimal Hamiltonian splitting

The HOLD integrator determines the accuracy of a kick between particles i and j from the particle-based time steps $\tau(i)$ and $\tau(j)$. In the CC integrator the accuracy of a kick is determined by the time step graph generated directly from interaction-based time steps $\tau(i, j)$. Could we further improve the splitting by applying kicks directly based on the interaction-specific time step criteria $\tau(i, j)$?

We implemented this idea in an experimental integrator which we named the OK split (OK stands for *Optimal Kick*). The method partitions a list of all *interactions* in the system (based on a pivot time step h) just like the HOLD split partitions a list of all *particles* in the system. The partitioning is formally optimal in the sense that every kick is evaluated at the time step closest to $\tau(i, j)$ in the power of two hierarchy based on the pivot time step h. While the possibility of direct N-body integration with interaction-based time steps has been previously considered in Nitadori & Makino (2008), the OK split is the first workable implementation of this idea that we are aware of.

In numerical tests, the OK split is not competitive compared to other methods such as the CC split. For example, in the 1024-body smoothed Plummer sphere test from Sect. 3.1, the relative energy error at the end of the simulation is around 10^{-2} (several orders of magnitude worse than HOLD and CC, but possibly still enough for drawing statistically correct conclusions, Portegies Zwart & Boekholt 2014). The remaining integrals of motion are conserved at machine precision, as the OK split applies kicks in pairs. Finally, the evolution of the mass distribution is comparable to HOLD and CC with the OK split using fewer kick and time step evaluations.

Could we improve the OK split by changing the time step criteria? For example, consider $\tau_\star(i, j) = \min(\tau(i), \tau(j))$ where τ is the particle-based time step criteria as defined in Pelupessy et al. (2012). Formally, combining the OK split with $\tau_\star(i, j)$ would result in a splitting with the exact same kicks and drifts as the HOLD integrator. This somewhat contrived example only serves the point of illustrating that the time step criteria can qualitatively change the behaviour of the OK split. While it is unknown whether practical interaction-based time step criteria even exist, we do believe that a closer look at the various simplifications made during the derivation of the explicit and approximately time-symmetric time step criteria that we have used throughout this work (Eq. (19)) would serve as a good starting point.

Acknowledgements. We thank Guilherme Gonçalves Ferrari and the anonymous referee for a critical reading of the manuscript. This work was supported by the Netherlands Research Council NWO (Grants #643.200.503, #639.073.803 and #614.061.608) and by the Netherlands Research School for Astronomy (NOVA). Jürgen Jänes was supported by the Archimedes Foundation, Estonian Students' Fund USA, Estonian Information Technology Foundation and Skype.

References

Ahmad, A., & Cohen, L. 1973, J. Comput. Phys., 12, 389
Barnes, J., & Hut, P. 1986, Nature, 324, 446
Bate, R. R., Mueller, D. D., & White, J. E. 1971, Fundamentals of Astrodynamics, Dover Books on Aeronautical Engineering Series (Dover Publications)
Belleman, R. G., Bédorf, J., & Portegies Zwart, S. F. 2008, New Astron., 13, 103
Capuzzo-Dolcetta, R., Spera, M., & Punzo, D. 2013, J. Comput. Phys., 236, 580
Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. 2001, Introduction to Algorithms, 2nd edn. (MIT Press)
Duan, W., Song, M., & Yates, A. 2009, BMC Bioinformatics, 10, S4
Gaburov, E., Harfst, S., & Portegies Zwart, S. 2009, New Astron., 14, 630

Gonçalves Ferrari, G., Boekholt, T., & Portegies Zwart, S. F. 2014, MNRAS, 440, 719
Goodwin, S. P., & Whitworth, A. P. 2004, A&A, 413, 929
Hairer, E., C., L., & Wanner, G. 2006, Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations (Springer Verlag)
Harish, P., & Narayanan, P. 2007, in High Performance Computing – HiPC 2007, eds. S. Aluru, M. Parashar, R. Badrinath, & V. Prasanna (Berlin, Heidelberg: Springer), Lect. Notes Comput. Sci., 4873, 197
Heggie, D. C., & Mathieu, R. D. 1986, in The Use of Supercomputers in Stellar Dynamics, eds. P. Hut, & S. L. W. McMillan (Berlin: Springer Verlag), Lect. Notes Phys., 267, 233
Konstantinidis, S., & Kokkotas, K. D. 2010, A&A, 522, A70
Makino, J. 1991, PASJ, 43, 859
McMillan, S. L. W. 1986, in The Use of Supercomputers in Stellar Dynamics, eds. P. Huts, & S. L. W. McMillan (Berlin: Springer Verlag), Lect. Notes Phys., 267, 156
Mikkola, S. 2008, in IAU Symp. 246, eds. E. Vesperini, M. Giersz, & A. Sills, 218
Nitadori, K., & Makino, J. 2008, New Astron., 13, 498
Oshino, S., Funato, Y., & Makino, J. 2011, PASJ, 63, 881
Pelupessy, F. I., Jänes, J., & Portegies Zwart, S. 2012, New Astron., 17, 711
Pelupessy, F. I., van Elteren, A., de Vries, N., et al. 2013, A&A, 557, A84
Portegies Zwart, S., & Boekholt, T. 2014, ApJ, 785, L3
Portegies Zwart, S., McMillan, S. L. W., van Elteren, E., Pelupessy, I., & de Vries, N. 2013, Comput. Phys. Commun., 183, 456
Portegies Zwart, S. F., Belleman, R. G., & Geldof, P. M. 2007, New Astron., 12, 641
Portegies Zwart, S. F., McMillan, S. L. W., & Gieles, M. 2010, ARA&A, 48, 431
Sanz-Serna, J. M., & Calvo, M. P. 1994, Numerical Hamiltonian problems, Applied Mathematics and Mathematical Computation 7 (London, New York: Chapman & Hall)