A&A 529, A27 (2011)
DOI: 10.1051/0004-6361/201014949

Astronomy & Astrophysics

SEREN – a new SPH code for star and planet formation simulations

Algorithms and tests⋆

D. A. Hubber[1,2,3,4], C. P. Batty[2], A. McLeod[2,5], and A. P. Whitworth[2]

[1] Department of Physics and Astronomy, University of Sheffield, Hicks Building, Hounsfield Road, Sheffield S3 7RH, UK
e-mail: D.Hubber@sheffield.ac.uk
[2] School of Physics and Astronomy, Cardiff University, Queens Buildings, The Parade, Cardiff, CF24 3AA, Wales, UK
[3] Institute for Theoretical Astrophysics, University of Oslo, Pb 1029 Blindern, 0315 Oslo, Norway
[4] Centre of Mathematics for Applications, University of Oslo, Pb 1053 Blindern, 0316 Oslo, Norway
[5] Astronomical Institute, Academy of Sciences of the Czech Republic, Boční II 1401, 141 31 Praha 4, Czech Republic

Received 6 May 2010 / Accepted 29 January 2011

ABSTRACT

We present SEREN, a new hybrid Smoothed Particle Hydrodynamics and *N*-body code designed to simulate astrophysical processes such as star and planet formation. It is written in Fortran 95/2003 and has been parallelised using OpenMP. SEREN is designed in a flexible, modular style, thereby allowing a large number of options to be selected or disabled easily and without compromising performance. SEREN uses the conservative "grad-h" formulation of SPH, but can easily be configured to use traditional SPH or Godunov SPH. Thermal physics is treated either with a barotropic equation of state, or by solving the energy equation and modelling the transport of cooling radiation. A Barnes-Hut tree is used to obtain neighbour lists and compute gravitational accelerations efficiently, and an hierarchical time-stepping scheme is used to reduce the number of computations per timestep. Dense gravitationally bound objects are replaced by sink particles, to allow the simulation to be evolved longer, and to facilitate the identification of protostars and the compilation of stellar and binary properties. At the termination of a hydrodynamical simulation, SEREN has the option of switching to a pure *N*-body simulation, using a 4th-order Hermite integrator, and following the ballistic evolution of the sink particles (e.g. to determine the final binary statistics once a star cluster has relaxed). We describe in detail all the algorithms implemented in SEREN and we present the results of a suite of tests designed to demonstrate the fidelity of SEREN and its performance and scalability.

Key words. hydrodynamics – methods: numerical – stars: formation

1. Introduction

Star formation problems are amongst the most demanding in computational astrophysics, requiring a large number of physical processes to be modeled (e.g. hydrodynamics, self-gravity, optically thick radiative cooling, gas chemistry, ionization, gas-ion coupling, magneto-hydrodynamics, radiative and mechanical feedback) over a very large range of physical conditions (i.e. gas densities from $\sim 10^{-20}\,\mathrm{g\,cm^{-3}}$ to $\sim 10^{+1}\,\mathrm{g\,cm^{-3}}$, and gas temperatures from ∼10 K to $\sim 10^{7}$ K). It is non-trivial to include all of the above physics in a single code which works over such a wide range of physical conditions and produces accurate results in an efficient manner. There are also often multiple methods available to model these processes, and one must choose the most appropriate and/or optimal method to study a given problem. The principal choice is whether to use an Eulerian grid-based code to simulate the fluid dynamics, or a Lagrangian particle-based code. Grid-based schemes are capable of modelling incompressible fluid dynamics accurately and efficiently, but for highly compressible fluids, where the density can take a large range of values, expensive adaptive-mesh-refinement techniques are required. Particle-based schemes, such as smoothed particle hydrodynamics, do not model hydrodynamical processes as well as grid-based schemes (Agertz et al. 2007), but they can model highly compressible fluids through a large range of scales with ease. This makes particle codes well suited to modelling self-gravitating fluids such as those involved in star formation. A number of publicly available codes using either static or adaptive-mesh-refinement grids (e.g. ZEUS, Stone & Norman 1992; FLASH, Fryxell et al. 2000; RAMSES, Teyssier 2002; ENZO, Abel et al. 2002) or particles (e.g. GADGET, Springel et al. 2001; GADGET2, Springel 2005; VINE, Wetzstein et al. 2009; Nelson et al. 2009) are available, and have been applied to a variety of different phenomena in interstellar gas dynamics, star and galaxy formation, and cosmology.

Here we present SEREN, a new multi-dimensional self-gravitating hydrodynamics and *N*-body code. SEREN uses the smoothed particle hydrodynamics (SPH) algorithm to model fluid dynamics, in combination with tree-gravity and hierarchical block-timestepping routines. It also includes a variety of specialist routines designed to tackle star and planet formation problems, such as sink particles (Bate et al. 1995), and a 4th order Hermite *N*-body integrator (Makino & Aarseth 1992) to follow the ballistic evolution of a star cluster once its gas has been accreted or dispersed.

The purposes of this paper are (i) to describe the algorithms implemented in SEREN, and (ii) to demonstrate the fidelity of SEREN – i.e. that the algorithms are coded correctly and reproduce known results in tests, so that future publications presenting

⋆ Further information and additional tests of SEREN can be found at the web-page http://www.astro.group.shef.ac.uk/seren.

simulations performed with SEREN can refer to this paper for a full description of the code. In Sect. 2, we give a brief overview of SEREN and all of its main features, and compare these features with those available in other available astrophysical SPH codes. In Sect. 3, we describe in detail the SPH algorithms used. In Sect. 4, we describe the implementation of self-gravity in SPH. In Sect. 5, we briefly discuss the available thermal physics modules, including the transport of heating, cooling and ionizing radiation. In Sect. 6, we discuss the integration schemes and time-stepping criteria. In Sect. 7, we discuss the implementation of sink particles. In Sect. 8, we discuss the 4th order Hermite N-body integrator and the additional features contained within it (e.g. binary identification). In Sect. 9, we discuss the implementation of the Barnes-Hut tree, and how it is used to determine neighbour lists and calculate gravitational accelerations. In Sect. 10, we present a large suite of tests, to demonstrate that SEREN simulates correctly the physical processes it is intended to capture. In Sect. 11, we discuss the memory optimisations used. In Sect. 12, we discuss the techniques used to parallelise SEREN using OpenMP, and we demonstrate how SEREN scales on shared-memory machines. In Sect. 13, we outline the major features that are still to be implemented.

2. Overview of SEREN and other codes

SEREN is a multi-dimensional self-gravitating SPH and N-body code. It has been designed for star and planet formation problems, but it can easily be adapted to simulate other astrophysical phenomena. SEREN is written in Fortran 95 (with some Fortran 2003 features) and is parallelised using OpenMP. It is written in a highly modular style, with a large number of features that can be switched on or off using Makefile options and conditional compilation flags. It can be compiled for one, two or three-dimensions, although it is most optimal in three-dimensional mode. We list here the main algorithms and features included in SEREN:

- standard SPH (e.g. Monaghan 1992), "grad-h" SPH (Springel & Hernquist 2002; Price & Monaghan 2004b), and Godunov SPH (Inutsuka 2002; Cha & Whitworth 2003);
- kernel-softened self-gravity (Price & Monaghan 2007b);
- artificial dissipation (Lattanzio & Monaghan 1985; Balsara 1995; Monaghan 1997; Morris & Monaghan 1997; Price 2008);
- 2nd-order Runge-Kutta, 2nd-order Predictor-Corrector and 2nd-order kick-drift-kick and drift-kick-drift Leapfrog integration schemes;
- hierarchical block time-stepping (e.g. Hernquist & Katz 1989);
- periodic boundary conditions, including periodic gravity (Hernquist et al. 1991; Klessen 1997);
- several particle types: self-gravitating gas particles, non-gravitating inter-cloud particles, static or non-static boundary particles;
- octal-spatial trees for neighbour-searching and gravity (Barnes & Hut 1986; Pfalzner & Gibbon 1996);
- simple isothermal, polytropic or barotropic equations of state, solution of the energy equation with associated radiation transport (Stamatellos et al. 2007), and propagation of ionizing radiation using HEALPix rays (Bisbas et al. 2009);
- sink particles (Bate et al. 1995);
- 4th order Hermite N-body integrator (Makino & Aarseth 1992);
- identification of binaries and calculation of binary properties (e.g. Aarseth 2003).

We control which algorithms are used in SEREN using Makefile options, so only the employed subroutines are compiled and included in the executable file. The parameters which determine how the selected algorithms function are set in a separate parameters file. In Sects. 3 to 9, we describe in more detail the implementation of these algorithms in SEREN.

Several other SPH codes are available to the astrophysics community for performing simulations of self-gravitating hydrodynamics. While these codes share a common set of basic features, most contain specialised algorithms to model certain astrophysical processes, or are optimised to perform a particular class of simulation. We briefly discuss the algorithms and features in other available astrophysical SPH codes, in order to highlight to potential users the relative merits of each code for solving particular problems and how they contrast with the features implemented in SEREN. We only discuss here those codes that have a refereed or archived publication containing details of the implementation and tests.

2.1. GADGET and GADGET 2

GADGET (Springel et al. 2001) and GADGET 2 (Springel 2005) are written in C and parallelised using MPI. While the original GADGET code was designed to investigate galaxy formation problems, GADGET 2 was designed to investigate large-scale cosmological problems such as galaxy cluster formation and the formation of structure in the Universe (e.g. Springel et al. 2005). MPI can be used very efficiently when the work distributed to all CPUS is automatically load-balanced. Therefore, the approximately uniform (large-scale) density distribution used in cosmological simulations is a problem that an MPI code like GADGET 2 can handle efficiently on very large clusters with 1000 s of CPUs (e.g. Springel 2005). GADGET 2 uses a Peano-Hilbert space-filling curve in order to determine how to distribute the particles amongst the available processors. This improves the scalability, by reducing communication overheads. GADGET 2 uses a conservative SPH formulation combined with solving the entropy equation for the thermal physics (Springel & Hernquist 2002). Particle properties can be integrated using either a Leapfrog-KDK or Leapfrog-DKD integration scheme, in combination with a hierarchical block-timestep scheme. The calculation of gravitational forces is split into short and long-range computations; short range forces are computed using a Barnes-Hut tree (which is efficient for clumpy density distributions), and long-range forces are computed using a particle-mesh scheme (which is efficient for smoother density distributions). GADGET 2 contains the ability to model several different particle types relevant to galaxy and cosmology simulations, namely gas, cold-dark matter and star particles. Star particles usually represent a whole cluster of stars, in comparison to sink particles in SEREN which represent individual stars, or unresolved small, multiple systems.

2.2. GASOLINE

GASOLINE (Wadsley et al. 2004) is written in Fortran and is parallelised for shared-memory machines. GASOLINE uses the standard formulation of SPH (e.g. Monaghan 1992) with (α, β) viscosity (Monaghan & Gingold 1983) and a Balsara switch (Balsara 1995) for reducing unwanted shear viscosity. GASOLINE can use two separate trees; a K-D tree for neighbour searching and a Barnes-Hut octal tree (Barnes & Hut 1986) for calculating gravitational forces. The code computes multipole moments up to hexadecapole-order to compute gravitational forces efficiently, but only uses the geometrical MAC for evaluating the cell-interaction list. Ewald summation is also available for simulating periodic boxes. GASOLINE contains a number of options for treating thermal physics, including an implicit integrator for solving the energy equation. A number of cooling and ionisation processes can be selected, as well as a simple heating-feedback prescription due to star formation. GASOLINE uses a Leapfrog-KDK integration scheme for advancing particle positions and velocities, along with a standard hierarchical block-timestep scheme.

2.3. VINE

VINE (Wetzstein et al. 2009; Nelson et al. 2009) is written in Fortran 95 and parallelised using OpenMP. As with GADGET and GADGET 2, VINE has been designed to investigate galaxy and cosmological problems. VINE has also been parallelised using OpenMP. It has been tested on up to 128 CPUs and scales well provided the problem size is large enough. VINE uses a nearest-neighbour binary tree (e.g. Benz et al. 1990) to compute gravitational forces and to search for neighbours efficiently. VINE also has the facility to use GRAPE boards (e.g. Makino et al. 2003) and thus can significantly speed up the calculation of the gravitational forces for particular problems. VINE does not use a conservative form of SPH, but rather uses the traditional form of SPH (Monaghan 1992). VINE contains a variety of particle types similar to GADGET 2, such as gas, cold-dark matter and star particles.

2.4. MAGMA

MAGMA (Rosswog & Price 2007) is an Smoothed Particle Magneto-hydrodynamics (SPMHD) code which is parallelised using OpenMP. MAGMA has been designed to model compact objects, such as binary-neutron stars. MAGMA uses the conservative "grad-h" SPH scheme for computing hydro and gravitational forces, and "Euler potentials" for solving the ideal MHD equations; this enforces div $\boldsymbol{B} = 0$ by design. The code includes dissipative artificial viscosity, conductivity and resistivity terms, with switches such as time-dependent viscosity (Morris & Monaghan 1997) for reducing dissipation. Thermal physics includes an equation-of-state for modelling the nucleon fluid at super-nuclear densities, and a method for modelling neutrino emission. No additional particle types (e.g. sink particles) are included. Gravitational forces are computed with a binary tree (Benz et al. 1990) and particle-particle interactions are computed with kernel-softened gravity (Price & Monaghan 2007b). Particle positions and velocities are integrated with a second-order predictor-corrector scheme, using an individual timestep scheme.

2.5. EvoL

EvoL (Merlin et al. 2010) is written in Fortran 95 and is parallelised using MPI. EvoL was designed to investigate cosmological structure, galaxy-cluster and galaxy formation problems, similar to GADGET 2 and VINE. As with other galaxy/cosmological codes, EvoL can model self-gravitating gas, cold dark matter and star particles. EvoL models the gas-dynamics using a modified "grad-h" SPH formulation, and also contains terms that correct for unequal-mass particles. Gravity is calculated using a Barnes-Hut tree, and neighbouring particle gravitational forces are computed with a conservative scheme similar to Price & Monaghan (2007b), but using the number density instead of the mass density, which again is beneficial when using unequal mass particles. EvoL uses a Leapfrog-KDK scheme for integrating particle positions and velocities. A standard hierarchical block-timestep scheme is employed, along with the instantaneous timestep-reduction procedure (Saitoh & Makino 2009) to ensure the timesteps used for neighbouring particles are not greatly different. EvoL also contains the ability to evolve the particle positions using the X-SPH method (e.g. Monaghan 1992) which can prevent particle interpenetration.

3. Smoothed particle hydrodynamics

SPH is a Lagrangian hydrodynamics scheme which uses particles to represent the fluid (Gingold & Monaghan 1977; Lucy 1977). SEREN contains three different variants of SPH: the standard implementation (Monaghan 1992), the conservative "grad-h" implementation (Springel & Hernquist 2002; Price & Monaghan 2004b) and the Godunov implementation (Cha & Whitworth 2003; Eqs. (9), (10)). The "grad-h" implementation is the favoured, default implementation in SEREN.

In SPH, particle properties are smoothed over a length scale, h, called the *smoothing length*, using a weighting function, $W(\boldsymbol{r}, h)$, called the *kernel function*. The fluid is thus still a continuous medium despite being represented by a finite number of discrete particles. The volume over which a particle is smoothed is called its *smoothing kernel*. Particle i interacts hydrodynamically with all other SPH particles, j, that lie inside the smoothing kernel of i (*gather*), and/or whose own smoothing kernels overlap i (*scatter*). These particles are referred to as the *neighbours* of i. The smoothing length determines the spatial resolution and can in principle be set to any value. The simplest choice is to keep h uniform in space and constant in time, throughout the simulation. However, to take advantage of the Lagrangian nature of SPH, it is often desirable to set the smoothing length of an SPH particle to be of order the local mean particle separation. The resolution then automatically adapts to the local conditions, providing an adaptability that is much more difficult to achieve with grid codes. SEREN contains two choices for the kernel function, both of which have finite

extent, $r_{\rm MAX} = \mathcal{R}h$: the M4 cubic spline kernel (Monaghan & Lattanzio 1985) with $\mathcal{R} = 2$, and the quintic spline kernel (Morris 1996) with $\mathcal{R} = 3$. Detailed properties of these kernels are given in Appendix A.

Since "grad-h" is the default implementation of SPH in SEREN, we briefly describe its main features here. In order to guarantee conservation of momentum, angular momentum and energy, the SPH fluid equations are derived from the Euler-Lagrange equations. This requires that the smoothing length of a particle be either constant, a function of the particle's co-ordinates, or a function of some property that is itself a function of the particle's co-ordinates. We follow Springel & Hernquist (2002) and Price & Monaghan (2004b) in making the smoothing length a function of the density. Specifically, for particle i we put

$$h_i = \eta_{\rm SPH}\left(\frac{m_i}{\rho_i}\right)^{\frac{1}{D}}, \tag{1}$$

where m_i is the mass of particle i, ρ_i is the SPH density at the position of particle i, D is the spatial dimensionality, and $\eta_{\rm SPH}$ is a parameter that controls the mean number of neighbours, $\bar{N}_{\rm NEIB} \simeq 2\mathcal{R}\eta_{\rm SPH}$, $\pi(\mathcal{R}\eta_{\rm SPH})^2$, $(4\pi/3)(\mathcal{R}\eta_{\rm SPH})^3$ in one, two and three dimensions respectively. ρ_i is calculated using

$$\rho_i = \sum_{j=1}^{N} m_j W(\boldsymbol{r}_{ij}, h_i), \tag{2}$$

where $\boldsymbol{r}_{ij} \equiv \boldsymbol{r}_i - \boldsymbol{r}_j$, and the summation includes particle i itself. Since the smoothing length is needed in order to calculate the density in Eq. (2) and vice versa in Eq. (1), h_i and ρ_i are obtained by iteration.

Once h and ρ are evaluated for all particles, the terms in the SPH fluid equations can be computed. The momentum equation is

$$\frac{{\rm d}\boldsymbol{v}_i}{{\rm d}t} = -\sum_{j=1}^{N} m_j\left(\frac{P_i}{\Omega_i\rho_i^2}\nabla_i W(\boldsymbol{r}_{ij}, h_i) + \frac{P_j}{\Omega_j\rho_j^2}\nabla_i W(\boldsymbol{r}_{ij}, h_j)\right), \tag{3}$$

where P_i is the pressure of particle i, $\nabla_i W$ is the gradient of the kernel function at the position of particle i, and

$$\Omega_i = 1 - \frac{\partial h_i}{\partial\rho_i}\sum_{j=1}^{N} m_j\frac{\partial W}{\partial h}(\boldsymbol{r}_{ij}, h_i). \tag{4}$$

Ω_i is a dimensionless quantity that corrects for the spatial variability of h. $\partial h_i/\partial\rho_i$ is obtained explicitly from Eq. (1). $\partial W/\partial h$ is obtained from the kernel function (see Appendix A). The SPH energy equation is

$$\frac{{\rm d}u_i}{{\rm d}t} = \frac{P_i}{\Omega_i\rho_i^2}\sum_{j=1}^{N} m_j\boldsymbol{v}_{ij}\cdot\nabla W_{ij}(\boldsymbol{r}_{ij}, h_i), \tag{5}$$

where $\boldsymbol{v}_{ij} \equiv \boldsymbol{v}_i - \boldsymbol{v}_j$. Since the mass of each particle is constant, and the density is computed using Eq. (2), there is no need to solve the SPH continuity equation.

The summations in Eqs. (2)–(5) are formally over all particles in the simulation. However, since the kernels used in SEREN both have finite extent, the summations are actually only over the neighbours of particle i. SEREN uses a Barnes-Hut tree (Barnes & Hut 1986) to obtain neighbour lists. The procedures for constructing and walking the tree are described in Sect. 9.

3.1. Artificial viscosity and conductivity

In most formulations of SPH, artificial viscosity terms are needed to ensure that shocks are captured, i.e. that converging particle streams do not interpenetrate, but rather form a contact discontinuity, and that kinetic energy is converted into thermal energy at the shock, thereby generating entropy. SEREN includes two different forms of artificial viscosity: the standard (α,β) formulation (Monaghan & Gingold 1983), and the formulation based on Riemann solvers (Monaghan 1997). The Monaghan-Riemann formulation is the default in SEREN, and involves adding the following extra terms to the momentum and energy equations,

$$\left(\frac{{\rm d}\boldsymbol{v}_i}{{\rm d}t}\right)_{\rm DISS} = \sum_{j=1}^{N}\frac{m_j}{\overline{\rho}_{ij}}\left\{\alpha v_{\rm SIG}\boldsymbol{v}_{ij}\cdot\hat{\boldsymbol{r}}_{ij}\right\}\overline{\nabla_i W}\left(\boldsymbol{r}_{ij}, h_i, h_j\right), \tag{6}$$

$$\left(\frac{{\rm d}u_i}{{\rm d}t}\right)_{\rm DISS} = -\sum_{j=1}^{N}\frac{m_j}{\overline{\rho}_{ij}}\left\{\frac{\alpha v_{\rm SIG}(\boldsymbol{v}_{ij}\cdot\hat{\boldsymbol{r}}_{ij})^2}{2} + \alpha' v'_{\rm SIG}(u_i - u_j)\right\}\hat{\boldsymbol{r}}_{ij}\cdot\overline{\nabla_i W}\left(\boldsymbol{r}_{ij}, h_i, h_j\right), \tag{7}$$

where α and α' are user specified coefficients of order unity, $v_{\rm SIG}$ and $v'_{\rm SIG}$ are signal speeds, $\hat{\boldsymbol{r}}_{ij} = \boldsymbol{r}_{ij}/|\boldsymbol{r}_{ij}|$, and

$$\overline{\nabla_i W}(\boldsymbol{r}_{ij}, h_i, h_j) = \frac{\nabla_i W(\boldsymbol{r}_{ij}, h_i) + \nabla_i W(\boldsymbol{r}_{ij}, h_j)}{2}. \tag{8}$$

This form of artificial dissipation is chosen as the default because (a) it has a physically informed motivation; and (b) it can be generalised to model dissipation in other quantities while giving just as good results as the standard (α, β) viscosity when modelling

shocks. The dissipation term on the right-hand side of Eq. (6) and the first term on the right-hand side of Eq. (7) represent artificial viscosity – i.e. exchange of momentum between neighbouring particles which are approaching or receding from one another, and conversion of the kinetic energy lost into thermal energy – and they are moderated by the signal speed $v_{\rm SIG} = c_i + c_j - \boldsymbol{v}_{ij} \cdot \hat{\boldsymbol{r}}_{ij}$. The second term on the right-hand side of Eq. (7) represents artificial conductivity, and acts to smooth out gradients in the specific internal energy. For purely hydrodynamic simulations, Price (2008) advocates that the artificial conductivity be moderated by the signal speed

$$v'_{\rm SIG} = \sqrt{\frac{|P_i - P_j|}{\overline{\rho}_{ij}}}. \tag{9}$$

However, in self-gravitating simulations this can drain thermal energy from dense condensations, thereby artificially accelerating gravitational contraction. Wadsley et al. (2006) have proposed the alternative signal speed

$$v'_{\rm SIG} = |\boldsymbol{v}_{ij} \cdot \hat{\boldsymbol{r}}_{ij}| \tag{10}$$

for artificial conductivity. Both Eqs. (9) and (10) are included as options in SEREN. We note that when the Godunov-SPH formulation is selected, we can disable the artificial *viscosity* since the Riemann solver should allow us to capture shocks accurately. We may need to retain the artificial *conductivity* since our simple implementation of a Riemann solver into SPH does not address that problem.

3.1.1. Artificial viscosity switches

Artificial viscosity can have undesirable side effects. In the absence of shocks it can lead to kinetic energy being dissipated at an unacceptably high rate, i.e. much faster than would happen with physical viscosity; this is an important consideration in simulations of interstellar turbulence. It can also deliver an unacceptably high shear viscosity, and thereby corrupt shear flows; this is an important consideration in simulations of the long-term evolution of accretion discs. A number of devices has been proposed to reduce the artificial viscosity in regions where it is not needed. Three such viscosity limiters are included in SEREN. The first is the switch proposed by Balsara (1995) in which α is multiplied by the dimensionless quantity $\frac{1}{2}(f_i + f_j)$, where

$$f_i = \frac{|\nabla \cdot \boldsymbol{v}|_i}{|\nabla \cdot \boldsymbol{v}|_i + |\nabla \times \boldsymbol{v}|_i + 0.001\, c_i/h_i}. \tag{11}$$

In regions of strong compression (i.e. shocks), the $|\nabla \cdot \boldsymbol{v}|$ terms tend to dominate over the $|\nabla \times \boldsymbol{v}|$ term, so $f_i \rightarrow 1$. In regions where vorticity dominates (i.e. shear flows), the $|\nabla \times \boldsymbol{v}|$ term dominates, so $f_i \rightarrow 0$.

The second device (which can be used in conjunction with the first) is time-dependent viscosity (Morris & Monaghan 1997). In time-dependent viscosity, each particle i has its own value of α_i, which evolves according to the equation

$$\frac{{\rm d}\alpha_i}{{\rm d}t} = \frac{\alpha_{\rm MIN} - \alpha_i}{\tau_i} + S_i. \tag{12}$$

Here $\alpha_{\rm MIN}$ is the default value of α_i, and τ_i is the e-folding time on which α_i relaxes to $\alpha_{\rm MIN}$, if the source term,

$$S_i = {\rm MAX}\{-(\nabla \cdot \boldsymbol{v})_i, 0\}\,(\alpha_{\rm MAX} - \alpha_i), \tag{13}$$

vanishes (cf. Rosswog et al. 2000). Reasonable results are obtained with $\alpha_{\rm MIN} = 0.1$, since a small residual artificial viscosity is needed to suppress high-frequency particle noise. The e-folding time is given by $\tau_i = C\, h_i/c_i$ with $C \sim 5$ (i.e. roughly a sound-crossing time for the smoothing kernel). The source term ensures that if particle i enters a shock, α_i quickly increases towards $\alpha_{\rm MAX} \sim 1$, but as soon as the shock is passed it decays back to $\alpha_{\rm MIN}$. If we use (α, β) viscosity, then we set $\beta_i = 2\alpha_i$.

The third device is the pattern-matching switch described by Cartwright & Stamatellos (2010). This switch is very effective in pure Keplerian discs, i.e. non-self-gravitating equilibrium discs modelled in the frame of reference of the central star, but has not yet been adapted to work in more general situations.

4. Self-gravity

Although the calculation of gravitational accelerations resembles an N-body problem, with forces between point masses, one should – for consistency with the calculation of SPH accelerations – take proper account of fact that the underlying density field, given by Eq. (2), is actually continuous, and the gravitational potential is related to this continuous density field by Poisson's equation, $\nabla^2\Phi = 4\pi G\rho$. Price & Monaghan (2007b) derive the equations of self-gravitating SPH by including the gravitational potential in the Lagrangian, and then proceeding as in Price & Monaghan (2004a). It is then necessary to introduce two additional kernel functions, the gravitational acceleration kernel (ϕ') and the gravitational potential kernel (ϕ, called the softening kernel by Price & Monaghan 2007b),

$$\phi'(\boldsymbol{r}, h) = \frac{4\pi}{r^2} \int_0^r W(\boldsymbol{r}', h)\, r'^2\, {\rm d}r', \tag{14}$$

$$\phi(\boldsymbol{r}, h) = 4\pi \left(-\frac{1}{r} \int_0^r W(\boldsymbol{r}', h)\, r'^2\, {\rm d}r' + \int_0^r W(\boldsymbol{r}', h)\, r'\, {\rm d}r' - \int_0^{\mathcal{R}h} W(\boldsymbol{r}', h)\, r'\, {\rm d}r' \right). \tag{15}$$

As with the basic kernel function, W, and its other derivatives, these new gravitational kernels are computed in advance, on a grid, and stored, so that subsequently values can be obtained efficiently by interpolation. The forms of both of these kernels are discussed in Appendix A. Using these kernels, Price & Monaghan (2007b) show that the gravitational acceleration of particle i is

$$\left(\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t}\right)_{\mathrm{GRAV}} = -G\sum_{j=1}^{N} m_j\,\overline{\phi'}(\boldsymbol{r}_{ij},h_i,h_j)\,\hat{\boldsymbol{r}}_{ij} - \frac{G}{2}\sum_{j=1}^{N}\left\{\frac{\zeta_i}{\Omega_i}\nabla W_i(\boldsymbol{r}_{ij},h_i) + \frac{\zeta_j}{\Omega_j}\nabla W_i(\boldsymbol{r}_{ij},h_j)\right\}, \tag{16}$$

where

$$\overline{\phi'}(\boldsymbol{r}_{ij},h_i,h_j) = \frac{\phi'(\boldsymbol{r}_{ij},h_i)+\phi'(\boldsymbol{r}_{ij},h_j)}{2}, \tag{17}$$

$$\zeta_i = \frac{\partial h_i}{\partial h_j}\sum_{j=1}^{N} m_j\frac{\partial\phi}{\partial h}(\boldsymbol{r}_{ij},h_i), \tag{18}$$

and Ω_i is given by Eq. (4). The two summation terms in Eq. (16) are, respectively, the kernel-softened gravitational acceleration, and the "grad-h" corrections that account for adaptive smoothing lengths. The ζ_i term is calculated and stored when other SPH quantities are calculated (i.e. ρ_i, $(\nabla\cdot\boldsymbol{v})_i$, Ω_i, etc.). To compute ζ_i requires $\partial\phi/\partial h$, which can be calculated and stored, once the form of W has been specified (see Appendix A). The gravitational potential at the position of particle i due to all other particles is

$$\Phi_i = G\sum_{j=1}^{N} m_j\,\overline{\phi}(\boldsymbol{r}_{ij},h_i,h_j), \tag{19}$$

where

$$\overline{\phi}(\boldsymbol{r}_{ij},h_i,h_j) = \frac{\phi(\boldsymbol{r}_{ij},h_i)+\phi(\boldsymbol{r}_{ij},h_j)}{2}. \tag{20}$$

If we choose standard SPH or Godunov SPH, the second summation in Eq. (16) is omitted, and the total energy is not as well conserved (see Price & Monaghan 2007b).

To compute gravitational accelerations exactly, using Eqs. (16)–(18), requires a summation over all particle pairs and is therefore an $O(N^2)$ process. To speed up the computation of gravitational accelerations, SEREN uses a Barnes-Hut tree (Barnes & Hut 1986). The resulting gravitational accelerations are not exact, but the resulting small fractional errors are considered acceptable, since there are other comparable or larger sources of error. The implementation of the gravity tree is described in Sect. 9.

4.1. Periodic gravity

Cosmological simulations (e.g. Springel et al. 2005) and simulations of turbulent molecular clouds (e.g. Klessen et al. 2000) often set out to model a representative piece of an infinite (or much more extended) medium, by assuming that the infinite medium consists of an infinite number of replicas of the main computational domain, extending periodically in all directions, and then employing periodic boundary conditions. For purely hydrodynamic simulations, periodic wrapping is sufficient to give acceptable boundary conditions. When self-gravity is invoked, we must include a contribution to the acceleration from all the replicas of the computational domain, extending to infinity. SEREN does this using the Ewald method (Hernquist et al. 1991; Klessen 1997). If the computational domain is a cube of side-length L, the total gravitational acceleration exerted on particle i by all of the infinite replicas of particle j (but not directly by the particle j itself) is

$$\left(\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t}\right)_{\mathrm{EWALD,j}} = G\,m_j\left(\boldsymbol{f}(\boldsymbol{r}_{ij}) + \frac{\boldsymbol{r}_{ij}}{|\boldsymbol{r}_{ij}|^3}\right), \tag{21}$$

where

$$\boldsymbol{f}(\boldsymbol{r}) = -\sum_{\boldsymbol{n}}\frac{\boldsymbol{r}-\boldsymbol{n}L}{|\boldsymbol{r}-\boldsymbol{n}L|^3}\left\{\mathrm{erfc}\,(\alpha|\boldsymbol{r}-\boldsymbol{n}L|) + \frac{2\alpha}{\sqrt{\pi}}|\boldsymbol{r}-\boldsymbol{n}L|\exp\left(-\alpha^2|\boldsymbol{r}-\boldsymbol{n}L|^2\right)\right\} - \frac{1}{L^3}\sum_{\boldsymbol{k}}\frac{4\pi\boldsymbol{k}}{k^2}\exp\left(-\frac{k^2}{4\alpha^2}\right)\sin(\boldsymbol{k}\cdot\boldsymbol{r}) \tag{22}$$

and $\alpha = 2/L$. The first summation in Eq. (22) is over all replicas in all directions (i.e. all $\boldsymbol{n}$-space) and the second summation is over all phase-space (i.e. all $\boldsymbol{k}$-space). The summations converge rapidly and can be truncated for $|\boldsymbol{r}-\boldsymbol{n}L| < 3.6L$ and $k^2 < 40\pi^2/L^2$. SEREN computes the dimensionless correction forces for a wide range of separations and tabulates the values in a look-up table.

5. Thermal physics

SEREN contains several equation-of-state (EOS) algorithms which can be selected using Makefile options. In all cases we assume that the gas is ideal, and so the pressure and specific internal energy are related by

$$P = \frac{\rho k_{\mathrm{B}}T}{\overline{m}} = (\gamma-1)\rho u, \tag{23}$$

where $k_{\rm B}$ is Boltzmann's constant, $\overline{m}$ is the mean gas-particle mass, and γ is the ratio of specific heats. With Options 1 to 3 below there is no need to solve the SPH energy equation, whereas with Options 4 and 5 there is.

1. Isothermal equation of state. If the gas is isothermal at temperature $T_{\rm o}$,

$$P = c_{\rm o}^2 \rho, \tag{24}$$

with constant isothermal sound speed, $c_{\rm o} = (k_{\rm B} T_{\rm o}/\overline{m})^{1/2}$.

2. Polytropic equation of state. The polytropic EOS has the form

$$P = K \rho^{\eta} \tag{25}$$

where K is the polytropic constant and η is the polytropic exponent; the polytropic index is $n = (\eta - 1)^{-1}$.

3. Barotropic equation of state. SEREN includes a barotropic equation of state of the form

$$T = T_{\rm o} \rho \left\{ 1 + \left(\frac{\rho}{\rho_{\rm crit}} \right)^{\gamma - 1} \right\}. \tag{26}$$

This mimics the behaviour of interstellar gas in molecular clouds, where the gas is optically thin to its cooling radiation and approximately isothermal (at $T_{\rm o} \sim 10\,{\rm K}$) when the density is low ($\rho < \rho_{\rm crit} \sim 10^{-13}\,{\rm g\,cm^{-3}}$), and optically thick to its own cooling radiation and approximately adiabatic (e.g. with $\gamma \simeq 5/3$) at higher densities ($\rho > \rho_{\rm crit}$).

4. Adiabatic equation of state. We integrate the internal energy equation explicitly (using Eq. (5)) and then calculate the thermal pressure from Eq. (23). Changes in the specific internal energy are solely due to compressional and/or viscous heating.

5. Radiative cooling. The method of Stamatellos et al. (2007) is used to capture realistically the main effects of radiative heating and cooling (in the optically thin, thick and intermediate regimes), but without the expense of a full radiative transfer calculation. This algorithm uses local functions of state (namely the density, temperature and gravitational potential) to compute an approximate optical depth to infinity, and hence to obtain an approximate cooling rate. This cooling rate is then used to solve the energy equation implicitly, and hence to determine the thermal evolution of the gas.

6. Ionising radiation. SEREN also includes the option to model a single discrete source of ionising radiation (i.e. an OB star or tight cluster of OB stars) using the algorithm of Bisbas et al. (2009). This algorithm generates an isotropic distribution of HEALPix rays, which are split into smaller child rays wherever finer resolution is needed. The rays propagate until they reach the ionisation front, where they are terminated. Particles well inside the HII region are given a high temperature (~10 000 K) and particles well outside the HII region are treated with one of the EOS algorithms listed above. There is a region with thickness of order the local smoothing length in which the temperature variation is smoothed, so as to avoid problems associated with abrupt temperature discontinuities.

6. Time integration

6.1. Integration schemes

SEREN offers a choice of four integration schemes: 2nd-order Runge-Kutta, 2nd-order Leapfrog (kick-drift-kick *and* drift-kick-drift) and 2nd-order Predictor-Corrector. The default choice is the 2nd-order Leapfrog drift-kick-drift:

$$\boldsymbol{r}_i^{n+1/2} = \boldsymbol{r}_i^n + \boldsymbol{v}_i^n \frac{\Delta t}{2}, \tag{27}$$

$$\boldsymbol{v}_i^{n+1/2} = \boldsymbol{v}_i^n + \boldsymbol{a}_i^{n-1/2} \frac{\Delta t}{2}, \tag{28}$$

$$u_i^{n+1/2} = u_i^n + \dot{u}_i^{n-1/2} \frac{\Delta t}{2}, \tag{29}$$

$$\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^n + \boldsymbol{a}_i^{n+1/2} \Delta t, \tag{30}$$

$$\boldsymbol{r}_i^{n+1} = \boldsymbol{r}_i^n + \frac{1}{2}(\boldsymbol{v}_i^n + \boldsymbol{v}_i^{n+1}) \Delta t, \tag{31}$$

$$u_i^{n+1} = u_i^n + \dot{u}_i^{n+1/2} \Delta t. \tag{32}$$

The main advantage of this scheme is that it only requires one acceleration calculation per timestep, as opposed to two in the 2nd-order Runge-Kutta scheme. Leapfrog schemes (both the Leapfrog kick-drift-kick and drift-kick-drift) are *symplectic* (i.e. they conserve phase-space) and so they are more stable for orbital integration (for example, in disc simulations). They are also, in principle, time-reversible for constant, global timesteps. The use of block time-stepping breaks exact time-reversibility (see Sect. 6.3), and also breaks exact momentum and angular momentum conservation. The other integration schemes are included because some perform better than the Leapfrog scheme in non–self-gravitating problems, and to allow comparison with other codes that use different integrators.

6.2. Optimal timesteps

SEREN calculates (but does not explicitly use) the *optimal* timestep for particle i, Δt_i, by determining the minimum value of three separate timesteps. The first is based on a modified Courant condition of the form

$$\Delta t_{\rm COUR} = \gamma_{\rm COUR} \frac{h_i}{(1+1.2\alpha)c_i + (1+1.2\beta)h_i|\nabla \cdot \boldsymbol{v}|_i}. \quad (33)$$

The denominator contains the $h_i|\nabla \cdot \boldsymbol{v}|_i$ term (which is frame-independent) instead of the absolute speed, $|\boldsymbol{v}|_i$ (which is normally used in the Courant condition). The terms involving α and β in the denominator account for particles that are in the vicinity of shocks. The second timestep condition is an acceleration condition similar to those used in some N-body codes, i.e.

$$\Delta t_{\rm ACCEL} = \gamma_{\rm ACCEL} \sqrt{\frac{h_i}{|\boldsymbol{a}|_i + \eta_{\rm a}}} \quad (34)$$

where $\eta_{\rm a}$ is a small positive acceleration to ensure the denominator does not at any time fall to zero. The third timestep condition is the heating condition, which limits the fractional change in the internal energy per timestep,

$$\Delta t_{\rm ENERGY} = \gamma_{\rm ENERGY} \frac{u_i}{|{\rm d}u/{\rm d}t|_i + \eta_{\dot{u}}}, \quad (35)$$

where $\eta_{\dot{u}}$ is a small positive heating rate to ensure the denominator does not fall to zero. This timestep criterion is only used when the SPH energy equation (Eq. (5)) is solved explicitly. If we solve the energy equation implicitly (e.g. Stamatellos et al. 2007), we only use the Courant and acceleration timesteps, Eqs. (33) and (34), to compute the optimal timestep for particle i, Δt_i.

6.3. Hierarchical block timesteps

SEREN uses hierarchical block time-stepping (e.g. Aarseth 2003) to reduce the run-time of a simulation. In a typical star formation simulation, only a small fraction of the particles might require very small timesteps, for example those passing through a shock or those near the centre of a condensation. If a global timestep is used, accelerations are recalculated for all particles, irrespective of whether the recalculation is really needed. Instead, we allow each particle to have its own timestep, chosen from a binary hierarchy of possible values, $\Delta t_n = 2^n\,\Delta t_{\rm MIN}$, where $n = 0, 1, 2, ..., n_{\rm MAX}$. Particle i is then allocated the largest value of Δt_n from this hierarchy that is smaller than its optimal timestep, Δt_i (based on Eqs. (33)–(35)). By restricting the ratio of timesteps to integer powers of 2, we ensure that the particles are always synchronised at the end of the largest timestep, $\Delta t_{\rm MAX} = 2^{n_{\rm MAX}}\,\Delta t_{\rm MIN}$.

The acceleration of a particle is then recalculated with a frequency determined by its allocated timestep, Δt_n. The most expensive parts of this recalculation are those associated with walking the trees. At any time, the positions, velocities and thermodynamic properties of particles whose accelerations do not yet need to be recalculated are simply estimated by extrapolation.

The timestep for a particle is recalculated at the end of its current timestep, using Eqs. (33) to (35). When the allocated timestep of a particle decreases (i.e. it moves to lower n in the hierarchy), there is no problem, because any lower timestep in the hierarchy is automatically synchronised with the higher one from which the particle is descending. On the other hand, this is not necessarily the case when a particle's allocated timestep increases (i.e. it moves to higher n in the hierarchy). In this situation, we have to check that the lower timestep is correctly synchronised with the higher one before we can move the particle up (i.e. increase its allocated timestep). In addition, we only allow a particle to increase its allocated timestep one level at a time.

As shown by Saitoh & Makino (2009), SPH can perform poorly when neighbouring particles have very different timesteps. For example, in a high Mach-number shock, the particles may interpenetrate because particles from the low-density pre-shock gas have much longer timesteps than those in the high-density post-shock gas, and therefore in a single timestep they advance deep into the shocked region. SEREN mitigates this effect by broadcasting each particle's allocated timestep to all it neighbours. If one of the neighbours j of particle i has an allocated timestep which is more than two levels higher in the hierarchy (i.e. more than a factor 4 longer; $t_j > 4\,t_i$), the neighbour's timestep is automatically reduced to $t_j = 4\,t_i$ as soon as the timestep hierarchy is correctly synchronised.

7. Sink particles

Sink particles are used in SPH to allow simulations of star formation to be followed longer (Bate et al. 1995). Gravitational collapse inevitably leads to high densities, short smoothing lengths, high accelerations, and therefore short timesteps. Under these circumstances, even the use of block time-stepping (Sect. 6.3) cannot prevent run-times from becoming impractically long. To circumvent this problem, we replace dense condensations with sink particles. A sink particle possesses the collective properties of the condensation it represents (i.e. mass, centre-of-mass position and net momentum) but does not retain any information about the internal structure and evolution of the condensation. Thus SPH particles that would otherwise have continued evolving inexorably towards higher density (thereby using up ever increasing amounts of CPU-time) are instead excised from the simulation. This means that the dynamics of the remaining more diffuse gas, and the formation of additional condensations, can be followed in an acceptable run-time. The assumption is made that – in the absence of feedback from the resulting protostar – the only important effect that the material inside a sink particle will have on its surroundings is due to its gravitational field. Thus sink particles interact gravitationally, but not hydrodynamically, with other sink and SPH particles.

A sink particle is created when an SPH particle satisfies all the stipulated sink-creation criteria. These criteria are divided into default criteria and optional criteria. The SPH particle which triggers the formation of a sink is referred to as the seed particle. The

default criteria for sink creation are then (i) that the SPH density of the seed particle is greater than $\rho_{\rm SINK}$; and (ii) that there is no other sink particle within $2r_{\rm SINK}$ of the seed particle (i.e. a sink particle should not be formed overlapping a pre-existing sink particle). In principle, $\rho_{\rm SINK}$ and $r_{\rm SINK}$ can be chosen independently. However, the results are only realistic if the material going initially into a sink particle is resolved. The default procedure in SEREN is to set $r_{\rm SINK} = \mathcal{R}h_{\rm s}$, where $h_{\rm s}$ is the smoothing length of the seed particle. This means that different sink particles have slightly different radii. The option exists in SEREN to prescribe a universal $r_{\rm SINK}$.

The four optional sink-creation criteria are (iii) that the mean density of the seed particle and all its neighbours exceeds $\rho_{\rm SINK}$ (this ensures that a stochastic density fluctuation does not result in the formation of a sink); (iv) that the SPH velocity divergence of the seed particle is negative, $(\nabla \cdot \boldsymbol{v})_{\rm s} < 0$ (this ensures that the particles going into the sink are condensing, and not being sheared apart); (v) that the SPH acceleration divergence of the seed particle is negative, $(\nabla \cdot \boldsymbol{a})_{\rm s} < 0$ (this ensures that the condensation is not being torn apart by tidal forces); (vi) that the total mechanical energy of the seed particle and its neighbours (kinetic plus gravitational potential energy in the centre-of-mass frame) is negative.

Only one sink particle can be created in any one timestep; otherwise the possibility would exist to generate multiple overlapping sinks. At each timestep, SEREN loops over all the SPH particles, and finds those whose SPH density (or, if required, mean density) exceeds $\rho_{\rm SINK}$. These candidate seed particles are then ordered in a list of decreasing SPH density (or mean density), and SEREN runs through this list until it finds a seed particle that satisfies all the creation criteria, and creates a sink particle out of this seed particle and all its neighbours.

An SPH particle i is accreted by an existing sink particle s if (a) the SPH particle lies inside the sink-particle's radius, $|\boldsymbol{r}_{\rm i} - \boldsymbol{r}_{\rm s}| \leq r_{\rm SINK}$, and (b) the kinetic plus gravitational energy of the two-body system comprising the sink-particle and the SPH-particle is negative. The SPH particle's mass, linear and angular momentum are then assimilated by the sink particle, and the SPH particle itself is removed from the simulation. When determining which SPH particles are accreted by which sink particles, we first compile a list of all the SPH particles which are to be accreted by each sink, and only when these lists are complete do we update the sink properties (mass, position, momentum) to account for the SPH particles it has just assimilated. This is necessary because otherwise the accretion process would depend on the order in which the SPH particles were interrogated.

8. *N*-body integrator

Simulations of star formation very often result in the formation of multiple stellar systems. Such simulations are modelled with hydrodynamical codes until most of the gas has been accreted by protostars, or dispersed by feedback; this is referred to as the *accretion phase*. In the absence of magnetic fields and feedback, the accretion phase is driven entirely by the competition between thermal pressure, viscosity, and gravity. The system then enters the *ballistic phase*, in which N-body dynamics modify the final clustering and binary properties, typically over a period of several tens of crossing times (Van Albada 1968). SPH simulations are often terminated after the accretion phase and not evolved through the ballistic phase.

SEREN includes an N-body integrator, so that it can follow both the accretion phase and the ballistic phase, in a single simulation. SEREN switches from an SPH simulation of the accretion phase, to an N-body simulation of the ballistic phase, if one of two conditions are met: either the simulation has reached the end-time stipulated in the parameters file, or a critical fraction of the original gas mass has been accreted by sink particles. At the switch-over, SEREN identifies any SPH particles which are strongly bound to a particular sink. On the assumption that these SPH particles are either about to be accreted by that sink, or will form a tightly bound disc around it (and eventually be accreted or form a planetary system), they are instantaneously accreted by the sink to which they are bound. This ensures that their contribution to the overall gravitational potential is not suddenly lost at the switch-over.

One problem that corrupts N-body codes is inaccuracies resulting from close interactions. These can build up over the course of a simulation, or materialise quickly in near head-on interactions, causing large energy errors. A variety of techniques has been employed to alleviate this problem, such as using very short timesteps (e.g. Portegies Zwart et al. 2001), gravity-softening (Aarseth 2003) or transformation of the equations of motion (e.g. KS regularization, Stiefel & Scheifele 1971). In SEREN, the N-body code retains the kernel-softened gravity used in the SPH code, in order to ensure that the gravitational accelerations are computed consistently between the two parts of a simulation. This has the advantage of preventing large energy errors due to close interactions, but has the disadvantage of preventing the formation of close binaries (separations less than $r_{\rm SINK}$).

8.1. Hermite integrator

The N-body integrator of choice is a fourth-order Hermite integrator (Makino 1991; Makino & Aarseth 1992). The Hermite integrator has been presented in two different forms in the literature, either as a fourth-order leapfrog scheme or as a fourth-order predictor-corrector scheme (Aarseth 2003). SEREN uses the predictor-corrector version of the Hermite integrator. Both forms are considered superior to other 4th-order N-body integrators, in the sense of giving better energy conservation and allowing longer timesteps (Makino 1991; Aarseth 2003). The leapfrog version of the Hermite scheme also maintains many of the properties of a traditional 2nd order leapfrog integrator (for example, it is symplectic), but it is of higher order by virtue of using both the acceleration and its first time derivative.

The N-body code uses a global timestep informed by the Aarseth (2001) criterion,

$$\Delta t_i = \gamma \sqrt{\frac{|\boldsymbol{a}_i||\ddot{\boldsymbol{a}}_i| + |\dot{\boldsymbol{a}}_i|^2}{|\dot{\boldsymbol{a}}_i||\dddot{\boldsymbol{a}}_i| + |\ddot{\boldsymbol{a}}_i|^2}}. \tag{36}$$

Here $\dot{\boldsymbol{a}}$, $\ddot{\boldsymbol{a}}$ and $\dddot{\boldsymbol{a}}$ are, respectively, the 1st, 2nd and 3rd time derivatives of the acceleration, calculated at the end of the previous timestep; γ is an accuracy factor of order ~0.1 (Makino & Aarseth 1992). Next we calculate the acceleration and its time-derivative (sometimes called the *jerk*) at the beginning of the step. The acceleration is given by

$$\boldsymbol{a}_i^n = -G \sum_{j=1}^{N} m_j \overline{\phi'}(\boldsymbol{r}_{ij}, h_i, h_j)\, \hat{\boldsymbol{r}}_{ij}, \tag{37}$$

where $\overline{\phi'}$ is the same gravitational-acceleration kernel as used in calculating kernel-softened gravitational accelerations in SPH (Eq. (14)). The kernel-softening means we must account for the rate of change of the kernel function and include extra terms in the expression for the jerk (Makino & Aarseth 1992). Using the same notation as in Sect. 4, the expression for the jerk becomes

$$\dot{\boldsymbol{a}}_i^n = -G \sum_j \frac{m_j\, \overline{\phi'}(\boldsymbol{r}_{ij}, h_i, h_j)}{|\boldsymbol{r}_{ij}|} \boldsymbol{v}_{ij} + 3G \sum_j^N \frac{m_j\, (\boldsymbol{r}_{ij} \cdot \boldsymbol{v}_{ij})\, \overline{\phi'}(\boldsymbol{r}_{ij}, h_i, h_j)}{|\boldsymbol{r}_{ij}|^3} \boldsymbol{r}_{ij} - 4\pi G \sum_j^N \frac{m_j\, (\boldsymbol{r}_{ij} \cdot \boldsymbol{v}_{ij})\, \overline{W}(\boldsymbol{r}_{ij}, h_i, h_j)}{|\boldsymbol{r}_{ij}|^2} \boldsymbol{r}_{ij}. \tag{38}$$

The particle positions and velocities are then advanced to the end of the timestep,

$$\boldsymbol{r}_i^{n+1} = \boldsymbol{r}_i^n + \boldsymbol{v}_i^n \Delta t + \frac{1}{2}\boldsymbol{a}_i^n \Delta t^2 + \frac{1}{6}\dot{\boldsymbol{a}}_i^n \Delta t^3, \tag{39}$$

$$\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^n + \boldsymbol{a}_i^n \Delta t + \frac{1}{2}\dot{\boldsymbol{a}}_i^n \Delta t^2. \tag{40}$$

We calculate the acceleration and jerk again using the new positions and velocities. We can thus calculate the second and third time derivatives at the beginning of the step (Makino & Aarseth 1992),

$$\ddot{\boldsymbol{a}}_i^n = \frac{2\left(-3(\boldsymbol{a}_i^n - \boldsymbol{a}_i^{n+1}) - (2\dot{\boldsymbol{a}}_i^n + \dot{\boldsymbol{a}}_i^{n+1})\Delta t\right)}{\Delta t^2}, \tag{41}$$

$$\dddot{\boldsymbol{a}}_i^n = \frac{6\left(2(\boldsymbol{a}_i^n - \boldsymbol{a}_i^{n+1}) + (\dot{\boldsymbol{a}}_i^n + \dot{\boldsymbol{a}}_i^{n+1})\Delta t\right)}{\Delta t^3}. \tag{42}$$

Finally, we add the higher order terms to the position and velocity vectors,

$$\boldsymbol{r}_i^{n+1} = \boldsymbol{r}_i^{n+1} + \frac{1}{24}\ddot{\boldsymbol{a}}_i^n \Delta t^4 + \frac{1}{120}\dddot{\boldsymbol{a}}_i^n \Delta t^5, \tag{43}$$

$$\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^{n+1} + \frac{1}{6}\ddot{\boldsymbol{a}}_i^n \Delta t^3 + \frac{1}{24}\dddot{\boldsymbol{a}}_i^n \Delta t^4. \tag{44}$$

The values of $\boldsymbol{a}$, $\dot{\boldsymbol{a}}$, $\ddot{\boldsymbol{a}}$ and $\dddot{\boldsymbol{a}}$ computed at the end of the timestep allow the code to calculate the next time step using Eq. (36). This is not possible on the very first timestep, and there we use explicit equations to calculate $\ddot{\boldsymbol{a}}$ and $\dddot{\boldsymbol{a}}$ (e.g. Aarseth 2001, his Eqs. (6) and (7)); all subsequent timesteps are determined using $\ddot{\boldsymbol{a}}$ and $\dddot{\boldsymbol{a}}$ from Eqs. (41) and (42).

8.2. Identification of multiple systems

During the N-body simulation, SEREN automatically searches for binaries and hierarchical triples and quadruples. There is no single robust method for identifying a bound, stable multiple system that contains an arbitrary number of components. We use a simple two-stage procedure. The first stage is to identify all binary systems present at the current time. This involves calculating the two-body energies of all star-pairs in the simulation. If (a) stars 1 and 2 are found to be mutually most-bound (i.e. the two-body energy of stars 1 and 2 is a minimum and negative for both stars); and (b) stars 1 and 2 are not bound to any other stars (i.e. the two-body energies of 1 and 2 with all other stars are positive), then they are identified as a bound binary system. If the primary and secondary masses are m_1 and m_2 respectively, the instantaneous relative displacement is $\boldsymbol{r}_{12} \equiv \boldsymbol{r}_1 - \boldsymbol{r}_2$, and the instantaneous relative velocity is $\boldsymbol{v}_{12} = \boldsymbol{v}_1 - \boldsymbol{v}_2$, then the two-body energy and angular momentum are

$$E_\mathrm{b} = \frac{1}{2}\mu |\boldsymbol{v}_{12}|^2 + G m_1 m_2 \overline{\phi}(\boldsymbol{r}_{12}, h_1, h_2), \tag{45}$$

$$\boldsymbol{L} = \mu\, \boldsymbol{r}_{12} \times \boldsymbol{v}_{12}, \tag{46}$$

where $\mu = m_1 m_2/(m_1 + m_2)$. The orbital binary parameters are then given by

$$q = \frac{m_2}{m_1}, \tag{47}$$

$$a = -\frac{G m_1 m_2}{2E_\mathrm{b}}, \tag{48}$$

$$e = \left(1 - \frac{|\boldsymbol{L}|^2}{G\, a\, (m_1 + m_2)\, \mu^2}\right)^{1/2}. \tag{49}$$

The next stage is to search for hierarchical systems. In order to facilitate this search, each binary found in the previous step is replaced by a single ghost-binary particle. We then repeat the procedure performed in the first stage, searching for any mutually most-bound pairs, but now using the ghost-binaries and the remaining unattached stars. If a ghost-binary is found to be most-bound to a single star and vice versa, they are identified as an hierarchical triple, and the orbit of the system is calculated as above and recorded. If two ghost-binaries are found to be most-bound to each other, then they are recorded as an hierarchical quadruple.

9. Tree

SEREN uses an implementation of the Barnes-Hut tree (Barnes & Hut 1986; Pfalzner & Gibbon 1996) to rapidly obtain neighbour lists for SPH interactions, and to efficiently calculate gravitational accelerations. The Barnes-Hut tree is an octal-spatial decomposition tree that splits the volume of the tree cells at each level into eight equal-volume cubic sub-cells (or four equal-area square sub-cells in 2D) – recursively until only a few, or zero, particles remain in each sub-cell. The cells at which a branch of the tree terminates are called *leaf cells*. SEREN decomposes the particles as an ensemble, in a similar manner to the algorithm described by Pfalzner & Gibbon (1996) (as distinct from the original Barnes-Hut method, which considers one particle at a time as the tree structure is built). The Pfalzner & Gibbon algorithm makes it easier to parallelise the tree-build routine using OpenMP.

We construct two separate trees, one for particles that experience hydrodynamic accelerations and one for particles that experience gravitational accelerations. This is advantageous because these accelerations are computed using different cell properties. In the case where all SPH particles are self-gravitating, we can build the tree structure once, copy this structure to the second tree, but then stock the two trees (i.e. calculate the properties of the tree cells) separately. Since the timestep criteria restrict how far particles can move in any one timestep, the tree structure will not change appreciably from one timestep to the next. Therefore we only build the tree structure every ∼10 timesteps, but restock it every timestep.

9.1. Neighbour searching

The Barnes-Hut *neighbour* tree is constructed using all SPH particles. For each cell, we record (i) the position of centre of the bounding box containing all particles in the cell; (ii) the maximum distance of all particles from the bounding box centre; (iii) the maximum smoothing length of all particles in the cell, and, if it is a leaf cell; (iv) the identifiers of all particles contained in the cell. Storing these quantities enables us to find neighbours efficiently, either by gather (i.e. all particles for which $|\boldsymbol{r}_{ij}|^2 \leq \mathcal{R}^2 h_i^2$), or by scatter (i.e. all particles for which $|\boldsymbol{r}_{ij}|^2 \leq \mathcal{R}^2 h_j^2$), or both. When we perform a tree-search of this type, we obtain a *potential neighbour list*, which is guaranteed to contain all of the true neighbours but normally also contains non-neighbours. There is no need to cull this list, because all non-neighbours which are passed to the SPH routines have no effect, since the kernel and its derivatives are zero for non-neighbours.

9.2. Tree gravity

The Barnes-Hut *gravity* tree is built using only self-gravitating SPH particles. For each cell, we record by default (i) the total mass of the cell; (ii) the position of the centre of mass, and, if it is a leaf cell; (iii) the IDs of all SPH particles contained in the cell. Additionally, we can compute and store higher-order multipole terms, in order to calculate the gravity of a cell to greater accuracy. A multipole expansion can in principle be made up to any order, although it is usually optimal to truncate after only a few terms. The monopole term is simply the centre of mass term for each cell and the dipole term is always zero if calculated with respect to the centre of mass of the cell. In SEREN, we provide the option to include either the quadrupole moment terms, or the quadrupole and octupole moment terms. The equations for the quadrupole and octupole moment tensors of a cell are given in Appendix B. The quadrupole moment tensor is a traceless symmetric matrix, Q, meaning there are 5 independent terms to be stored for each cell. The octupole moment tensor is a more complicated rank-3 tensor, S, whose symmetries result in 10 independent terms which must be stored for each cell. The gravitational potential at the position of particle i due to cell c, up to octupole order, is

$$\phi_{\rm GRAV} = -\frac{GM_c}{|\boldsymbol{r}|} - \frac{GQ_{ab,c} r_a r_b}{2\,|\boldsymbol{r}|^5} - \frac{GS_{ab,c} r_a^2 r_b + GS_{123,c} r_1 r_2 r_3}{2\,|\boldsymbol{r}|^7} \tag{50}$$

where $\boldsymbol{r} = \boldsymbol{r}_i - \boldsymbol{r}_c$ is the position of particle i relative to cell c, (r_1, r_2, r_3) are the Cartesian components of $\boldsymbol{r}$, and we employ the Einstein summation convention (i.e. we sum over repeated indices). If we define $\hat{\boldsymbol{e}}_a$ to be the unit vector in the ath Cartesian direction, the gravitational acceleration of particle i, due to cell c, up to octupole order, is

$$\begin{aligned}\left(\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t}\right)_{\rm GRAV} = &-\frac{GM_c}{|\boldsymbol{r}|^3}\boldsymbol{r} + \frac{GQ_{ab,c} r_a}{2\,|\boldsymbol{r}|^5}\hat{\boldsymbol{e}}_b - \frac{5}{2}\frac{GQ_{ab,c} r_a r_b}{2\,|\boldsymbol{r}|^7}\boldsymbol{r} \\ &+\frac{GS_{ab,c} r_a r_b}{|\boldsymbol{r}|^7}\hat{\boldsymbol{e}}_a + \frac{GS_{ab,c} r_a^2}{|\boldsymbol{r}|^7}\hat{\boldsymbol{e}}_b - \frac{7GS_{ab,c} r_a^2 r_b}{|\boldsymbol{r}|^9}\boldsymbol{r} - \frac{7GS_{123,c} r_1 r_2 r_3}{2\,|\boldsymbol{r}|^9}\boldsymbol{r} + \frac{GS_{123,c}}{|\boldsymbol{r}|^7} \times (r_2 r_3 \hat{\boldsymbol{e}}_1 + r_3 r_1 \hat{\boldsymbol{e}}_2 + r_1 r_2 \hat{\boldsymbol{e}}_3)\,.\end{aligned} \tag{51}$$

When walking the gravity tree, the code must interrogate cells to decide whether to use the multipole expansion or to open up the cell and interrogate its child cells. This decision is determined by the *multipole-acceptance criterion* (MAC). SEREN includes a simple Geometric MAC and a GADGET-style MAC; it also includes a new Eigenvalue MAC, which uses the eigenvalues of the quadrupole moment terms to determine whether to open a cell.

9.2.1. Geometric MAC

The Geometric MAC uses the size of the cell, ℓ_c (i.e. its longest corner-to-corner length), and its distance from the particle, $|\boldsymbol{r}_i - \boldsymbol{r}_c|$, to calculate the angle the cell subtends at the particle, $\theta_{ci} = \ell_c/|\boldsymbol{r}_i - \boldsymbol{r}_c|$. If θ_c is smaller than some pre-defined tolerance, $\theta_{\rm MAC}$, the gravitational acceleration due to the cell is given by the multipole expansion, Eq. (51). If this criterion is not satisfied, the cell is opened and the sub-cells on the next level are interrogated in the same way. If a leaf cell is opened, we store the identifiers of all the particles contained in it, and compute their contribution to the net gravitational acceleration directly (using Eqs. (16)–(18)). For computational efficiency, the code calculates and stores for each cell the quantity $\mathcal{S}_c = (\ell_c/\theta_{\rm MAC})^2$. An unnecessary square-root operation is then avoided by applying the geometric MAC in the form

$$|\boldsymbol{r}_i - \boldsymbol{r}_c|^2 \geq \mathcal{S}_c \qquad \text{(CELL DOES NOT NEED TO BE OPENED)}. \tag{52}$$

9.2.2. GADGET-style MAC

Springel et al. (2001) have formulated another type of MAC for the SPH code GADGET. This MAC uses an approximation to the leading error term in the multipole expansion to calculate for each cell the smallest distance from the cell at which the multipole expansion can be used. GADGET includes quadrupole moment corrections, and so the leading error term is the octupole term. However, Springel et al. suggest that the octupole moment term is small in a homogeneous density field, in which case the hexadecapole term is the largest error term. For a cell c of total mass M_c and linear size ℓ_c, an approximation to the magnitude of the acceleration of particle i due to the hexadecipole term is $a_{\rm HEX} \sim GM_c\ell_c^4/|\boldsymbol{r}_i - \boldsymbol{r}_c|^6$. If $a_{\rm HEX}$ is less than some user-defined fraction of the total gravitational acceleration of particle i, i.e. $a_{\rm HEX} < \alpha_{\rm MAC}\,|\boldsymbol{a}_{\rm GRAV}|$, the multipole expansion is used; otherwise cell c must be opened to the next level. Since the current acceleration of particle i is not yet available, the code uses the acceleration from the previous timestep as an approximation. The code therefore calculates and stores for each cell the quantity $\chi_c = (GM_c\ell_c^4/\alpha_{\rm MAC})^{1/3}$, and then applies the GADGET-style MAC in the form

$$|\boldsymbol{r}_i - \boldsymbol{r}_c|^2 \geq \chi_c\,|\boldsymbol{a}_{\rm GRAV}|^{-1/3} \qquad \text{(CELL DOES NOT NEED TO BE OPENED)}. \tag{53}$$

When the quadrupole and octupole moment terms are not used, the leading error term is the quadrupole term. Therefore an approximation to the acceleration of the quadrupole term, $a_{\rm QUAD} = G\,M_c\ell_c^2/|\boldsymbol{r}_i - \boldsymbol{r}_c|^4$, is used instead. In this case the code calculates and stores for each cell the quantity $\chi'_c = (GM_c\ell_c^2/\alpha_{\rm MAC})^{1/2}$, and then applies the GADGET-style MAC in the form

$$|\boldsymbol{r}_i - \boldsymbol{r}_c|^2 \geq \chi'_c\,|\boldsymbol{a}_{\rm GRAV}|^{-1/2} \qquad \text{(CELL DOES NOT NEED TO BE OPENED)}. \tag{54}$$

Since we do not have a value of $\boldsymbol{a}_{\rm GRAV}$ on the very first timestep, we use the Geometric MAC with $\theta_{\rm MAC} = 1.0$ to obtain an initial estimate and then revert to Equations. (53) and (54). Equations (53) and (54) do not guarantee a maximum fractional force error, but rather attempt to set an upper limit on the error contribution from each cell. It is therefore possible that the error is larger than desired. Therefore we use the Geometric MAC with $\theta_{\rm MAC} = 1.0$, alongside the GADGET-style MAC, as a safety measure for the rare cases where Eqs. (53) and (54) are inadequate.

9.2.3. Eigenvalue MAC

We introduce here a new Eigenvalue MAC, based on the quadrupole moment terms of a cell. Salmon & Warren (1994) originally suggested using higher-order multipole moments directly to formulate a MAC, but they only used upper limits to the multipole moment terms to constrain the leading error term of the multipole expansion. This resulted in a more conservative MAC than was actually required to achieve the desired accuracy, and hence more expensive tree walks. The Eigenvalue MAC is formulated by determining the maximum values of the gravitational potential (or acceleration) due to the quadrupole moment terms of a cell. The quadrupole moment tensor is a real, symmetric and traceless matrix. It therefore has three real eigenvalues, $\lambda_1, \lambda_2, \lambda_3$. From Eq. (50), the gravitational potential due to the quadrupole moment term is

$$\phi_{\rm QUAD} = -\frac{G\,Q_{ab,c}r_a r_b}{2\,|\boldsymbol{r}|^5}. \tag{55}$$

The term in the numerator, $Q_{ab,c}r_a r_b$, is the *quadratic form* between the quadrupole matrix Q and the vector $\boldsymbol{r}$. It can be shown (e.g. Riley et al. 1997) that the quadratic form has a maximum absolute value given by $|\lambda_{\rm MAX}|\,|\boldsymbol{r}|^2$, where $\lambda_{\rm MAX}$ is the largest in magnitude of the three eigenvalues. We therefore solve the eigenvalue equation,

$$\det[Q - \lambda I] = A + B\lambda + C\lambda^2 - \lambda^3 = 0, \tag{56}$$

$$A = -Q_{33}Q_{12}^2 - Q_{22}Q_{13}^2 - Q_{11}Q_{23}^2 + 2Q_{12}Q_{13}Q_{23} + Q_{11}Q_{22}Q_{33} \equiv \det[Q], \tag{57}$$

$$B = Q_{12}^2 + Q_{13}^2 + Q_{23}^2 - Q_{11}Q_{22} - Q_{11}Q_{33} - Q_{22}Q_{33}, \tag{58}$$

$$C = Q_{11} + Q_{22} + Q_{33} \equiv \mathrm{Tr}[Q]. \tag{59}$$

Since, by design, $C = \mathrm{Tr}[Q] = 0$, Eq. (56) is a *depressed cubic equation*, i.e. a cubic equation with no quadratic term. Since also Q is real and symmetric, the eigenvalues are real, and Eq. (56) can be solved by the method of Vieta (e.g. Martin 1998). In particular, the largest eigenvalue is

$$\lambda_{\rm MAX} = \sqrt{\frac{4B}{3}} = 2\sqrt{\frac{Q_{12}^2+Q_{13}^2+Q_{23}^2-Q_{11}Q_{22}-Q_{11}Q_{33}-Q_{22}Q_{33}}{3}}. \tag{60}$$

Table 1. Initial conditions for the adiabatic Sod test (Cols. 2 and 3), and for the colliding flows test (Cols. 4 and 5).

ADIABATIC SOD TEST	$x < 0$	$x > 0$	ISOTHERMAL COLLIDING FLOWS	$x < 0$	$x > 0$
ρ	1.0	0.25		1.0	1.0
P	1.0	0.1795		1.0	1.0
v_x	0.0	0.0		4.0	−4.0

We therefore require that the magnitude of the quadrupole moment potential, $|\phi_{\rm QUAD}| = G\lambda_{\rm MAX}/2|\boldsymbol{r}_i - \boldsymbol{r}_c|^3$, be less than some user-defined fraction of the total potential, $|\phi_{\rm QUAD}| < \alpha_{\rm MAC}|\phi_{\rm GRAV}|$. The code approximates $\phi_{\rm GRAV}$ with the value from the previous timestep, and calculates and stores for each cell the quantity $\xi_c = \{G^2(Q_{12}^2 + Q_{13}^2 + Q_{23}^2 - Q_{11}Q_{22} - Q_{11}Q_{33} - Q_{22}Q_{33})/3\alpha_{\rm MAC}^2\}^{1/3}$. The Eigenvalue MAC is then applied in the form

$$|\boldsymbol{r}_i - \boldsymbol{r}_c|^2 \geq \xi_c\,|\phi_{\rm GRAV}|^{-2/3} \qquad \text{(CELL DOES NOT NEED TO BE OPENED)}. \tag{61}$$

Equation (61) does not guarantee a maximum fractional error, but attempts to limit the error contribution from each cell. Therefore we also use the Geometric MAC with $\theta_{\rm MAC} = 1.0$, alongside the Eigenvalue MAC, as an extra safety measure.

9.2.4. SPH-neighbour cell-opening criterion

The multipole expansions used in SEREN assume that each SPH particle in a cell is a point-mass. In contrast, the derivation of the equation of motion takes account of the finite extent of an SPH particle (i.e. kernel-softened gravity; see Sect. 4). The SPH-neighbour criterion therefore requires that any cell that might contain neighbours of particle i be opened. The code calculates and stores for each cell the quantity $d_c^2 = \mathrm{MAX}_j\{(|\boldsymbol{r}_j - \boldsymbol{r}_c| + \mathcal{R}h_j)\}$, where the maximum is over all the particles j in the cell; d_c is the maximum extent of the smoothing kernels of the particles in cell c. The overall cell-opening criterion then takes the form

$$|\boldsymbol{r}_i - \boldsymbol{r}_c|^2 \geq \mathrm{MAX}\left\{(\mathcal{R}h_i)^2;\;\; d_c^2;\;\; \mathcal{S}_c \text{ OR } \chi_c|\boldsymbol{a}_{\rm GRAV}|^{-1/3} \text{ OR } \chi_c'|\boldsymbol{a}_{\rm GRAV}|^{-1/2} \text{ OR } \xi_c|\phi_{\rm GRAV}|^{-2/3}\right\} \text{(CELL DOES NOT NEED TO BE OPENED)}. \tag{62}$$

This additional criterion adds an extra overhead to calculating the properties of the cells, and also to the gravity walk, since there are now two cell-opening criteria to check. However, in highly clustered geometries such as those found in gravitational collapse problems, this extra check brings significant accuracy and speed benefits.

10. Tests

We have performed a large number of standard and non-standard tests to demonstrate that the algorithms in SEREN have been implemented correctly and perform well. It is not practical to test all possible combinations of the options available in SEREN, and we have therefore chosen tests which demonstrate the performance of particular algorithms. Where possible, we compare the test results with known analytic or semi-analytic solutions. Where an algorithm has been developed in another SPH code and the subroutine then imported into SEREN (e.g. the radiative cooling module of Stamatellos et al. 2007), or has been written directly into SEREN as an independent module (e.g. the HEALPix module for treating ionising radiation; Bisbas et al. 2009), the testing is not described here, and the interested reader is referred to the original paper.

10.1. Generation of initial conditions

The generation of initial conditions in SPH often needs careful consideration, since particle noise and edge effects can impact negatively on test simulations such as those described here. For example, random initial conditions suffer from Poisson-noise in the particle distribution which leads to high-frequency noise in the density and particle accelerations. A safer approach is to generate a so-called "glass" distribution of particles. A glass is a semi-regular structure in which all the particles are roughly equidistant from each other.

In order to generate a glass, we initially place equal-mass particles randomly in a periodic box. The particles are evolved using SEREN, with artificial viscosity to dissipate the kinetic energy, until the particles have settled into an equilibrium structure. We use an isothermal EOS and a Courant factor of $\gamma_{\rm COUR} = 0.2$. Once settled, the particle boxes can be replicated and joined together to create larger settled particle distributions, and uniform-density spheres can be cut from a box. All of the glass distributions used in this paper are set-up using this method. We note that glass-structures can be set up with different methods (e.g. "repulsive" gravity in GADGET2, Springel 2005).

10.2. Adiabatic Sod test (Sod 1978)

The initial conditions for this test are summarised in Table 1 (left side). The computational domain is $-4 \leq x \leq +4$, $0 \leq y \leq 1$, $0 \leq z \leq 1$, and periodic wrapping is invoked in all three dimensions. Initially (at $t = 0$), the left-hand half of the domain ($x < 0$) contains a high-density, high-pressure gas, represented by 64 000 particles, and the right-hand half ($x > 0$) contains a low-density, low-pressure gas, represented by 16 000 particles. The particles have been relaxed to a glass, and are at rest; they have equal mass. The gas evolves adiabatically, with adiabatic exponent $\gamma = 1.4$. We therefore solve the momentum and energy equations, using both artificial viscosity and artificial conductivity, to moderate the discontinuities in velocity and temperature, respectively. We perform

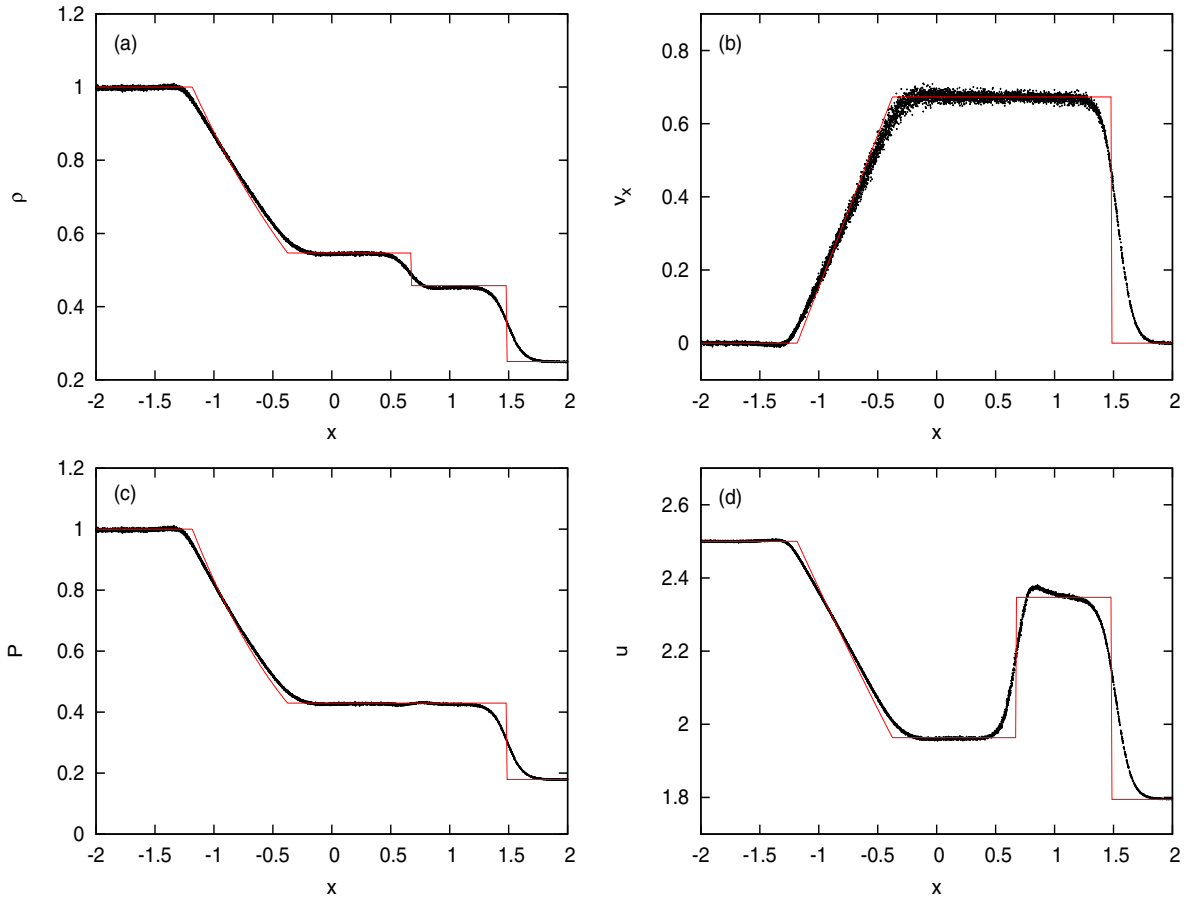

Fig. 1. Results of the adiabatic shock test using the "grad-h" SPH formulation (Price & Monaghan 2004a) showing **a)** the density, **b)** the x-velocity, **c)** the thermal pressure, and **d)** the specific internal energy after a time $t = 1.0$. The black dots represent the results from the SPH simulation and the red lines show the semi-analytic solution obtained using a Riemann solver.

this test in 3D (since this is the dimensionality of star-formation simulations) using the default "grad-h" SPH method, the Monaghan (1997) artificial viscosity and the Price (2008) artificial conductivity.

Figure 1 shows the density, x-velocity, thermal pressure and specific internal energy profiles (black dots), and the accurate 1D solution obtained using a Riemann solver (red lines), at the end of the simulation ($t = 1$) in the interval $|x| < 2$. A rarefaction wave is propagating into the high-density gas on the left (its head is at $x \sim -1.3$), and a shock wave is propagating into the low-density gas on the right (it has reached $x \sim 1.5$). There is also a contact discontinuity (at $x \sim 0.6$), since the gas from the right has higher specific entropy than that from the left. The SPH results reproduce the gross features of the accurate solution well, but the discontinuities are inevitably spread over a few smoothing lengths.

10.3. Colliding flows test

The initial conditions for this test are summarised in Table 1 (right side). The computational domain is $-4 \leq x \leq +4$, $0 \leq y \leq 0.2$, $0 \leq z \leq 0.2$, and periodic wrapping is invoked in the y and z dimensions, but not in the x dimension. Initially, the density is uniform, but the gas in the left-hand half of the computational domain ($x < 0$) has velocity $v_x = +4$, and the gas in the right-hand half ($x > 0$) has velocity $v_x = -4$; the gas is represented by 128 000 equal-mass particles which have been relaxed to a glass. The velocities are smoothed near $x = 0$ instead of having an unresolved x-velocity discontinuity. Therefore the discontinuous velocity profile, $\boldsymbol{v}'(\boldsymbol{r})$, is replaced by the smoothed velocity,

$$\boldsymbol{v}_i = \sum_{j=1}^{N} \frac{m_j}{\rho_j} \boldsymbol{v}'_j \, W(\boldsymbol{r}_{ij}, h_i). \tag{63}$$

The gas is isothermal, with dimensionless sound speed $c_s = 1$, so the code does not need to solve the energy equation, nor does it need to invoke artificial conductivity. This test demonstrates how well artificial viscosity enables the code to suppress particle interpenetration and capture shocks. We perform the test in 3D, using standard SPH with Monaghan (1997) with and without time-dependent artificial viscosity (Morris & Monaghan 1997). For the time-dependent viscosity simulation, we set $\alpha_{\rm MAX} = 2$ (and $\beta_{\rm MAX} = 4$) with $\alpha_{\rm MIN} = 0.1$. We adopt a global timestep for both simulations.

Figure 2 compares the SPH density and x-velocity as a function of x (black dots) with the analytic solution (red line), for the standard SPH run, and Fig. 3 makes the same comparison for the time-dependent viscosity run, both at $t = 0.6$. The peak density and

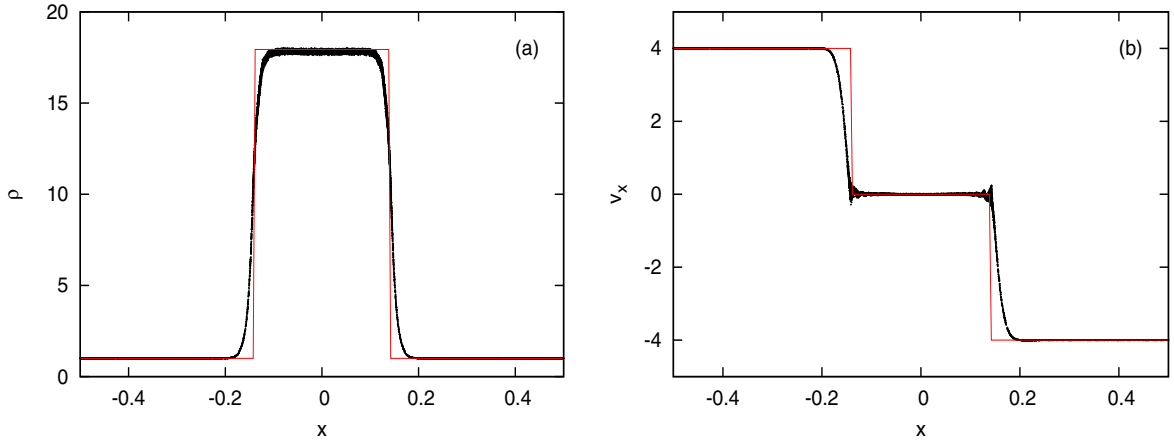

Fig. 2. Results of the colliding flows test using the standard SPH equations with the Monaghan (1997) artificial viscosity showing **a)** the density and **b)** the x-velocity, after a time $t = 0.6$. The black dots represent the results from the SPH simulation and the red lines show the analytic solution.

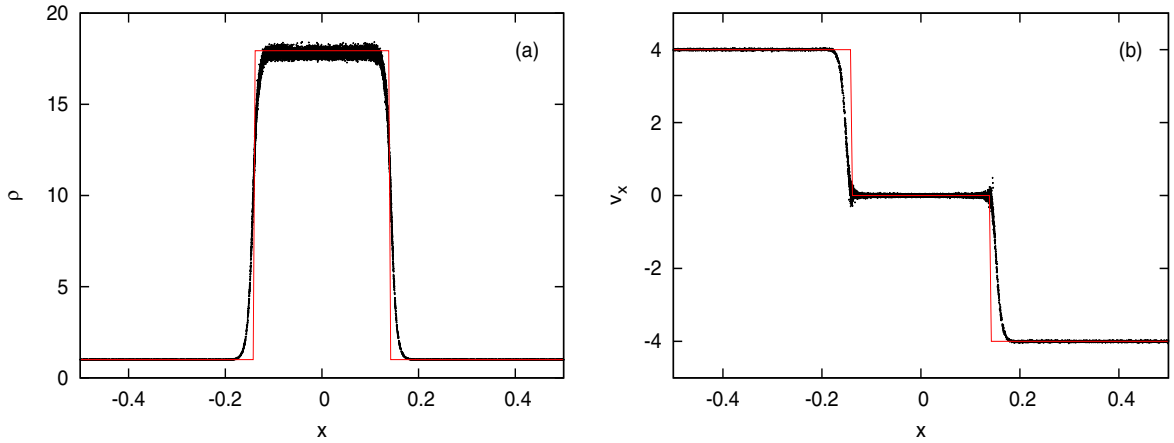

Fig. 3. Results of the colliding flows test using the standard SPH equations and time-dependent (α, β) viscosity (Morris & Monaghan 1997) showing **a)** the density and **b)** the x-velocity, after a time $t = 0.6$. The black dots represent the results from the SPH simulation and the red lines show the analytic solution.

the width of the shock are in agreement with the analytic solution for both runs, but the discontinuities in density and velocity are smeared out over a few smoothing lengths. This smearing is an inherent feature of SPH simulations. The time-dependent viscosity performs almost as well as the standard Monaghan (1997) viscosity, with a little more scatter in the post-shock density. The scatter in the density is partly a result of increased particle disorder at the shock front due to a small amount of particle penetration when the shock forms (Figs. 2b and 3b).

10.4. Sedov blast wave (Sedov 1959)

This test demonstrates that the code can handle the steep temperature and density gradients created by an explosion, and the consequent requirement for a timestep limiter (see Sect. 6.3 and Saitoh & Makino 2009). A settled, uniform-density glass-like distribution of 200 000 SPH particles is created. Then the central particle and its (~50) neighbours are given a net impulse of thermal energy $\Sigma U = 1$, divided amongst them according to the smoothing kernel. The remaining particles have a total thermal energy 10^{-6} times smaller than the particle with the maximum internal energy (i.e. the particle closest to the centre). The impulse of thermal energy results in an outward propagating shock front which sweeps the surrounding gas into a dense layer. Sedov (1959) provides an analytic similarity solution for the subsequent evolution of this system (strictly speaking, one in which the surrounding particles start with zero thermal energy).

We perform three realisations of this test, with three different time-stepping schemes. The resulting density profiles at time $t = 0.02$ are shown in Fig. 4, and compared with the semi-analytic solution. The SPH simulation with global timesteps (Fig. 4a) shows good agreement with the semi-analytic solution; the maximum density in the shell is reduced by smoothing, but the position and width of the shock front are comparable with the analytic solution. The SPH simulation using hierarchical block time-steps (Fig. 4b) fails to reproduce any of the features of the semi-analytic solution, because the cold particles have such a long timestep, compared with the hot ones, that they cannot respond to the pressure of the explosion and the hot particles penetrate through them. The SPH simulation using hierarchical block timesteps with a timestep limiter (i.e. not allowing any SPH particle to have a timestep more than four times longer than its neighbours; Fig. 4c) produces results which are indistinguishable from the simulation using global timesteps, but uses ~8% of the computing time.

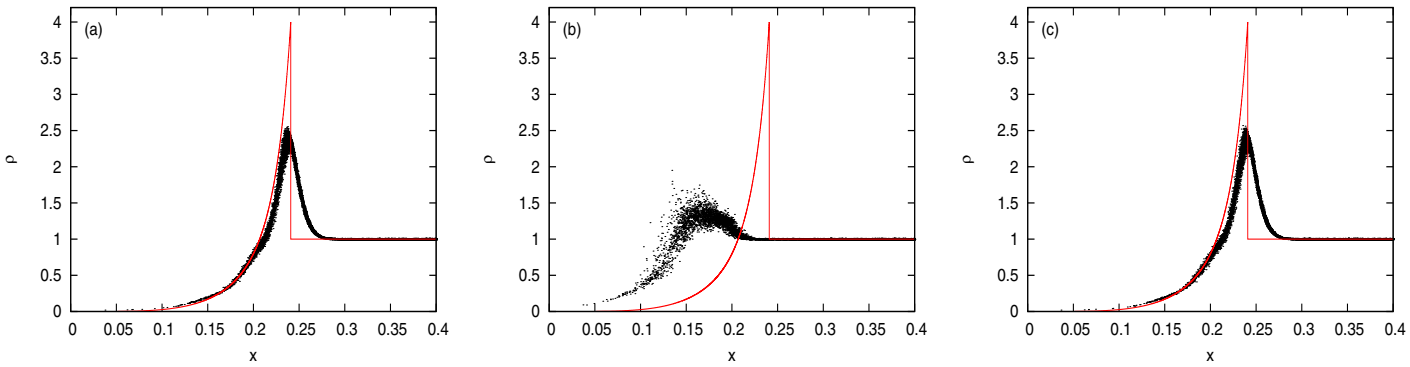

Fig. 4. Results of the Sedov blast wave test at a time $t = 0.02$ using **a)** global timesteps; **b)** individual block timesteps; **c)** individual timesteps with the timestep-limiter described in Sect. 6.3 and in Saitoh & Makino (2009). The block dots represent the SPH results and the red line shows the semi-analytic solution provided by Sedov (1959).

10.5. Kelvin-Helmholtz instability

The Kelvin-Helmholtz instability (hereafter KHI) is a classical hydrodynamical instability that occurs, in the simplest case, between two bulk flows that are shearing past one another. It has been extensively studied in recent years as a diagnostic for comparing the ability of both SPH and grid codes to model mixing of interacting fluids (e.g. Agertz et al. 2006; Price 2008; Read et al. 2010). In particular, this test has highlighted an intrinsic problem in the standard formulation of SPH and has led to several suggested modifications to SPH (e.g. Price 2008; Read et al. 2010).

We use similar initial conditions to Springel (2010) where two fluids with densities $\rho_1 = 1$ and $\rho_2 = 2$ are in shear-flow along the $y = 0$ plane with relative velocity $|v_1 - v_2| = 1.0$. The two fluids are in pressure-balance, $P = 2.5$, and have ratio of specific heats $\gamma = 5/3$. Therefore, there is discontinuity in the specific internal energy, $u = P/(\gamma - 1)\,\rho$, and also in the specific entropy. Both layers are contained within a periodic box of extent $-0.5 < x < 0.5$ and $-0.5 < y < 0.5$. Springel (2010) adds a velocity perturbation of the form

$$v_y(x, y) \;=\; w_0 \,\sin\!\left(\frac{2\,\pi}{\lambda}\right) \left\{ \exp\left[-\frac{(y - y_{I1})^2}{2\sigma^2} \right] + \exp\left[-\frac{(y - y_{I2})^2}{2\sigma^2} \right] \right\} \tag{64}$$

x where $\lambda = 0.5$ is the wavelength of the velocity perturbation between the two fluids, $w_0 = 0.1$ is its amplitude and $\sigma = 0.05/\sqrt{2}$ is the scale-height of the perturbation in the y-direction. We invoke time-dependent artificial viscosity and artificial conductivity (see Sect. 3.1). We adopt the quintic kernel (see Appendix A.2) for computing all SPH quantities, instead of the more common M4 kernel (see Appendix A.1). We follow the growth of the instability for a total dimensionless time of $t = 1.5$. The linear growth-timescale of the instability is

$$\tau_{\rm KH} \;=\; \frac{(\rho_1 + \rho_2)}{\sqrt{\rho_1\,\rho_2}} \frac{\lambda}{|\boldsymbol{v}_2 - \boldsymbol{v}_1|}. \tag{65}$$

For our initial conditions, the growth timescale is $\tau_{\rm KH} = 1.06$. Therefore, by the end of the simulation the instability should have entered the non-linear phase where significant vorticity and mixing occur near the shearing interface. We perform simulations using these initial conditions at both low (12 242 particles) and high (98 290 particles) resolutions.

In Fig. 5, we show the evolution of the density field and the development of the instability at three different times, $t = 0.5$, 1.0 and 1.5. For both the low and high resolution cases, the instability evolves at approximately the same rate through the linear-phase ($t = 0.5$), and subsequently during the non-linear phase ($t = 1.0$ and 1.5) where significant vorticity develops. The large-scale properties of the vortices formed are very similar in both the low and high resolutions cases. The main difference between the two is the number of resolved spiral turns in a vortex. The low resolution case has just enough spatial resolution to model the formation of one complete spiral loop by $t = 1.5$. The high resolution case has enough resolution to model two complete spiral loops and can be seen to have less dispersion in the density field around the contact regions between the two fluids.

10.6. Tree multipole expansion and scaling characteristics

We test the accuracy of the Barnes-Hut gravity tree and the multipole moment correction terms (Sect. 9.2), by comparing the gravitational acceleration obtained by walking the tree, $\boldsymbol{a}_i^{\rm TREE}$, with that obtained by a direct-summation over all particles, $\boldsymbol{a}_i^{\rm DIRECT}$ (cf. McMillan & Aarseth 1993). Specifically, we compute the root-mean-square fractional acceleration error,

$$\epsilon \;=\; \left(\frac{1}{N} \sum_{i=1}^{N} \left\{ \frac{\left| \boldsymbol{a}_i^{\rm TREE} - \boldsymbol{a}_i^{\rm DIRECT} \right|^2}{\left| \boldsymbol{a}_i^{\rm DIRECT} \right|^2} \right\} \right)^{1/2}. \tag{66}$$

The density field used in this test is a uniform-density, glass-like sphere (see Sect. 10.1) of 32 000 SPH particles; we note that this is actually a stiffer test of the tree than a highly structured density field. We compute ϵ using the Geometric MAC (Sect. 9.2.1) and

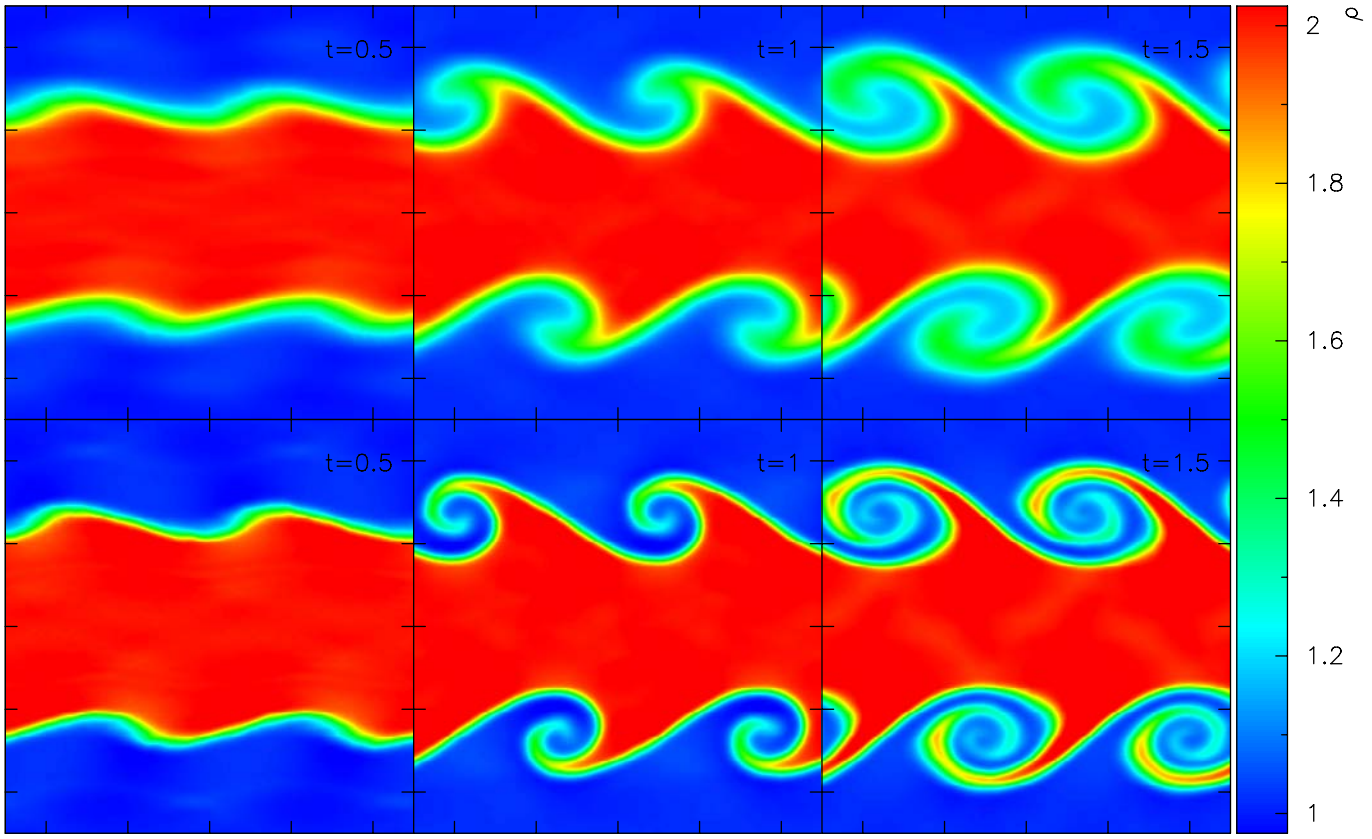

Fig. 5. Development of the Kelvin-Helmholtz instability for the low-resolution case (*top-row*; 12 242 particles) and the high-resolution case (*bottom-row*; 98 290 particles). The plots show the evolution of the density-field (colour bar on right-hand side) at times $t = 0.5$, 1.0 and 1.5 (*left, middle* and *right* columns respectively).

the Eigenvalue MAC (Sect. 9.2.3). For the Geometric MAC, we compute ϵ using different values of $\theta_{\rm MAC}$ in the range 0.1 to 1.0, and including terms up to monopole, quadrupole and octupole order. For the Eigenvalue MAC, we compute ϵ using different values of $\alpha_{\rm MAC}$ in the range 10^{-6} to 10^{-2}, and including terms up to quadrupole and octupole order; we do not consider monopole-only since we must calculate the quadrupole moment terms anyway in order to formulate the Eigenvalue MAC. We do not include the effects of kernel-softening in this test and therefore we effectively set the smoothing lengths to zero for the purposes of using the SPH-neighbour opening criterion; Sect. 9.2.4.

The resulting values of ϵ are plotted against $\theta_{\rm MAC}$ and $\alpha_{\rm MAC}$ in Figs. 6a and 7a. We see that for the Geometric MAC, ϵ decreases monotonically with decreasing $\theta_{\rm MAC}$ and with the inclusion of higher-order multipole terms (cf. McMillan & Aarseth 1993). Likewise the value of ϵ computed with the Eigenvalue MAC decreases monotonically for decreasing $\alpha_{\rm MAC}$.

In Figs. 6b and 7b, we plot the CPU time required to compute all the gravitational accelerations using, respectively, the Geometric and Eigenvalue MACs, against the computed RMS fractional force error, ϵ. For both MACS, and for all multipole-expansions, the CPU time increases as ϵ decreases. For the Geometric MAC, acceptably small values of ϵ are delivered much faster if the quadrupole terms are included. For both the Geometric and Eigenvalue MACs, the octupole terms do not deliver a big improvement in accuracy, and therefore – in the interests of memory and CPU efficiency – we normally evaluate only monopole and quadrupole terms.

The time required to calculate the gravitational accelerations using the tree is expected to scale as $N \log N$ (e.g. Pfalzner & Gibbon 1996), compared with N^2 for a direct-summation. Figure 8 shows the average CPU time for calculating gravitational accelerations in a uniform density sphere using the tree with the Geometric MAC (red triangles) and using direct-summation (solid black circles). The two graphs scale as expected up to 10^5 particles and beyond.

10.7. Freefall and isothermal collapse of a uniform density sphere

We test the accuracy of the gravitational acceleration evaluation in a dynamically-evolving system, by simulating the freefall collapse of a uniform density sphere. A static, uniform-density sphere with mass $M_{\rm o}$, initial radius $R_{\rm o}$ and initial density, $\rho_{\rm o} = 3M_{\rm o}/4\pi R_{\rm o}^3$, collapses to a singularity on a timescale $t_{\rm FF}$; and a shell of the sphere which is initially ($t = 0$) at radius $r_{\rm o}$ is at subsequent times ($0 < t \leq t_{\rm FF}$) at radius r, given by

$$\frac{t}{t_{\rm FF}} = \frac{2}{\pi}\left\{\cos^{-1}\left(\frac{r}{r_{\rm o}}\right)^{1/2} + \left(\frac{r}{r_{\rm o}}\right)^{1/2}\left(1-\frac{r}{r_{\rm o}}\right)^{1/2}\right\}, t_{\rm FF} = \frac{\pi}{2}\left(\frac{R_{\rm o}^3}{2\,G\,M_{\rm o}}\right)^{1/2} = \left(\frac{3\,\pi}{32\,G\,\rho_{\rm o}}\right)^{1/2}. \quad (67)$$

We set up the initial conditions by constructing a glass-like uniform-density sphere containing 100 000 SPH particles (as described in Sect. 10.1). The subsequent evolution of the particles is then followed invoking gravitational accelerations only. Figure 9a compares

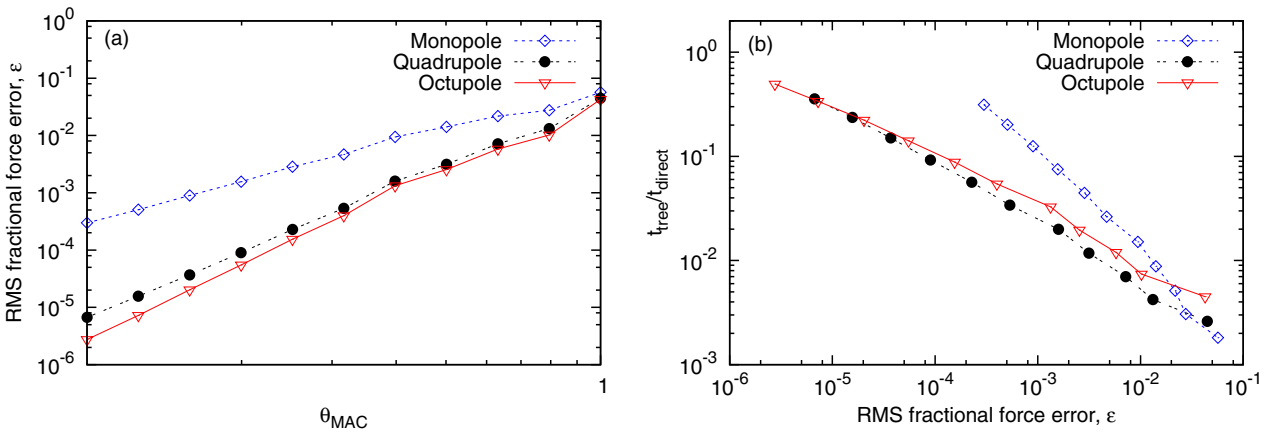

Fig. 6. **a)** The root-mean-square fractional force error computing the gravitational forces for all particles in a uniform-density sphere with the Barnes-Hut tree using the Geometric MAC as a function of $\theta_{\rm MAC}$, and **b)** the ratio of CPU time for computing all gravitational forces with the tree to direct-summation as a function ϵ. The gravitational accelerations are calculated without kernel-softening, up to monopole (blue diamonds), quadrupole (solid black circles) and octupole (red triangles) order.

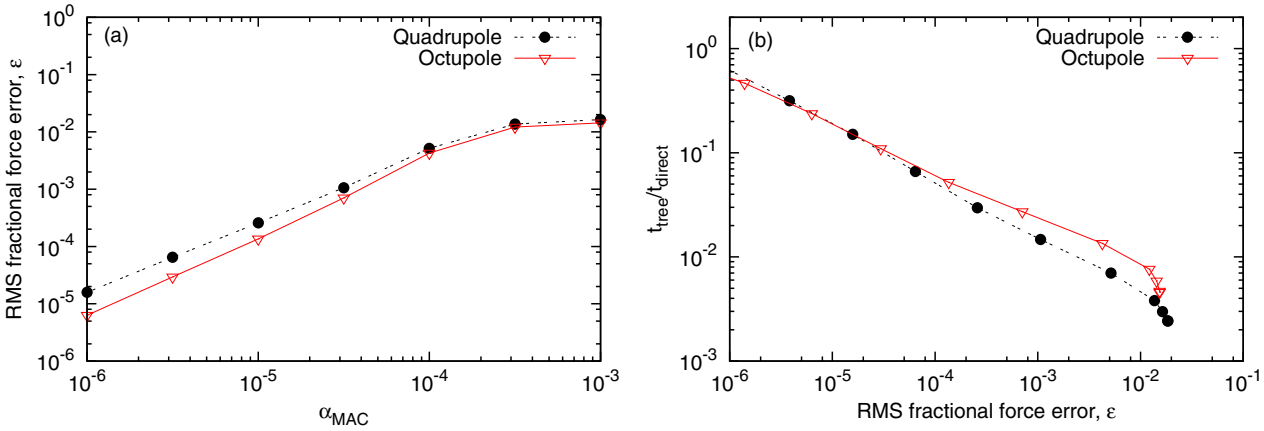

Fig. 7. **a)** The root-mean-square fractional force error computing the gravitational forces for all particles in a uniform-density sphere with the Barnes-Hut tree using the Eigenvalue MAC as a function of $\alpha_{\rm MAC}$, and **b)** the ratio of CPU time for computing all gravitational forces with the tree to direct-summation as a function ϵ. The gravitational accelerations are calculated without kernel-softening, up to quadrupole (solid black circles) and octupole (red triangles) order.

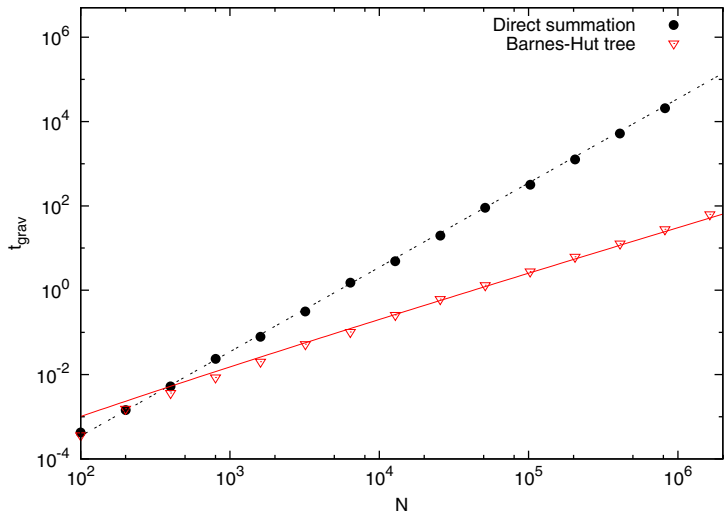

Fig. 8. Scaling characteristics of the Barnes-Hut tree code in SEREN. The time taken to compute gravitational forces for all particles for direct-summation (solid black circles) and the Barnes-Hut tree (red triangles) as a function of particle number. For reference, we show the expected scaling for N^2 (dashed line) and N log N (solid red line).

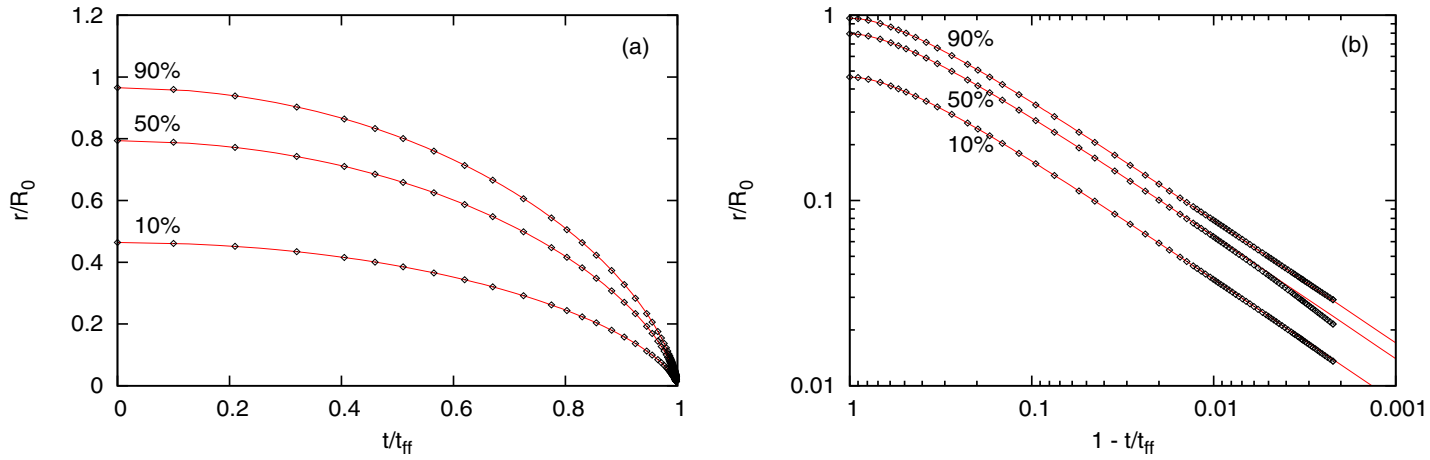

Fig. 9. **a)** Freefall collapse of a pressure-less, uniform-density sphere. The figure shows the analytic solution (dashed lines), and the radial position of three representative particles at 90%, 50% and 10% mass radii (filled circles). **b)** Same as **a)**, but with time measured from the end of the collapse and using logarithmic axis.

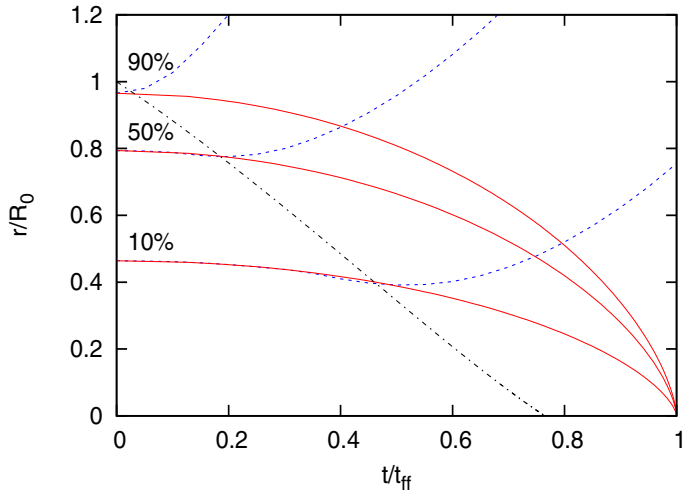

Fig. 10. Collapse of a uniform density sphere with an isothermal equation of state and dimensionless isothermal sound speed $c_s = 1$. The figure shows the analytic solution for pressure-less freefall collapse (solid red lines) and the radial position of the 90%, 50% and 10% mass radii (blue dashed lines). The black dot-dashed line shows the analytic solution for the progression of the rarefaction wavefront.

the 90%, 50% and 10% mass radii as a function of time (dots) with the analytic solution (dashed lines). Significant divergence between the numerical results and the analytic solution – due to gravitational softening, particle noise, and integration error – occurs only after the density has increased by more than 10^7 (see Fig. 9b).

This test has been repeated, but now imposing an isothermal equation of state and invoking both gravitational and hydrostatic accelerations. The collapse is no longer homologous, since there is a pressure gradient at the edge of the sphere, and this drives a rarefaction wave into the cloud. Ahead of the rarefaction wave, the gas collapses in freefall, as before, but behind it the gas decelerates and then expands. Figure 10 compares the 90%, 50% and 10% mass radii as a function of time (solid lines) with the analytic solution for the pressure-less collapse (Eq. 67; dashed lines) and the position of the rarefaction wave as a function of time (Truelove et al. 1998; dot-dashed line). We use a dimensionless isothermal sound speed, $c_s = 1$, so that the rarefaction wave reaches the centre of the sphere in less than a freefall time, preventing collapse to a singularity. Figure 10 shows that the gas motion diverges from freefall collapse just after the rarefaction wave passes, as it should. Slight deviations before this juncture are due to smoothing, gravitational softening, particle noise, and integration error.

10.8. Polytropes

This test demonstrates that SEREN can model the structure of an $\eta = 5/3$ polytrope, and therefore should be able to handle general self-gravitating equilibria. The density profile of a polytrope is obtained by solving the Lane-Emden equation (Chandrasekhar 1939, Chap. IV). An $\eta = 5/3$ polytrope with mass $M = 1$ and radius $R = 1$ (in dimensionless code units) has polytropic constant

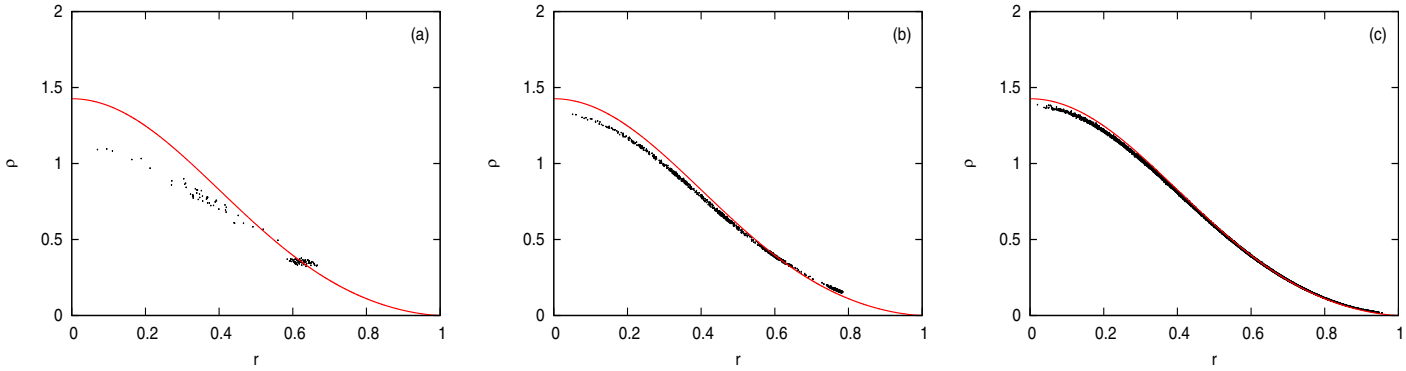

Fig. 11. Results of the polytrope test for an $\eta = 5/3$ polytrope, using **a)** 114, **b)** 1086, and **c)** 10^5 SPH particles.

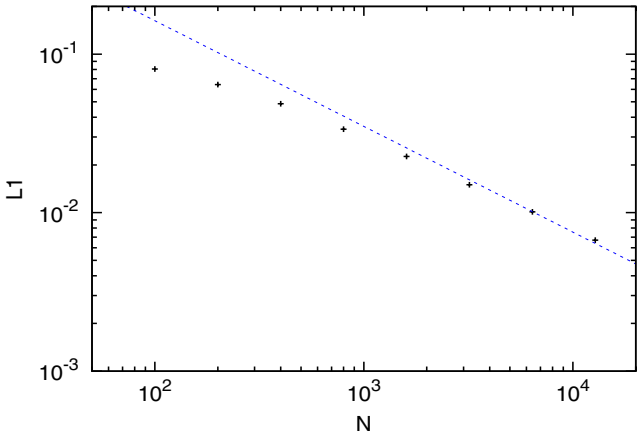

Fig. 12. L1 error norm as a function of particle number for the static polytrope test. The expected behaviour for an ideal 2nd-order numerical hydrodynamics scheme, $L1 \propto N^{-2/3}$, is also plotted for reference (blue dashed line).

$K = 0.4246$ (cf. Price & Monaghan 2007b). The initial conditions are generated by cutting a unit-mass, unit-radius sphere from a cube of settled particles (see Sect. 10.1), and then stretching the particles radially so that, in spherical polar co-ordinates, the new radius of particle i, r'_i, is related to its old radius, r_i, by $M_{\rm POLY}(r'_i) = r_i^3$, where $M_{\rm POLY}(r')$ is the mass interior to radius r' in the polytropic configuration; the angular co-ordinates of particle i are not changed. Stretching distorts the local arrangement of individual particles, and so the new configuration is not in detailed equilibrium. We therefore evolve it with $\eta_{\rm SPH} = 1.2$, using artificial viscosity, until the system reaches equilibrium. This test has been performed with $N = 114$, 1086 and 10^5 SPH particles. For $\eta_{\rm SPH} = 1.2$, $\bar{N}_{\rm NEIB} \simeq 57$. Bate & Burkert (1997) suggest that in SPH only condensations with $N \geq 2\bar{N}_{\rm NEIB}$ particles are resolved. Therefore our very low-resolution test with $N = 114 = 2\bar{N}_{\rm NEIB}$ particles (Fig. 11a) demonstrates that SEREN can indeed crudely model such a condensation, albeit with only approximately the correct radius and central density; the grouping of particles near the boundary (at $r \sim 0.6$) in Fig. 11a reflects the tendency for well relaxed distributions of SPH particles to adopt a glass-like arrangement. Figure 11b shows that with $N = 1086 \simeq 20\bar{N}_{\rm NEIB}$ the polytrope is much better resolved, and Fig. 11c shows that with $N = 10^5$ the density profile almost exactly matches the Lane-Emden solution.

We test convergence with the exact solution by calculating the L1 error norm as a function of particle number for varying resolutions. SPH is formally second-order accurate in space (e.g. Monaghan 1992) and therefore the L1 error should scale as $L1 \propto h^2 \propto N^{-2/D}$ where D is the dimensionality (cf. Springel 2010b). However, the discretization of the gas into particles introduces additional errors, so the error scales less well than second-order (e.g. $L1 \propto N^{-1} \log N$; e.g. Monaghan 1991). Figure 12 demonstrates the L1 error norm as a function of total particle number. We see that the L1 error norm decreases with increasing particle number, and therefore converges with increasing resolution. Also plotted in Fig. 12 is the expected scaling for an ideal second-order scheme. It can be seen that the convergence rate is similar to the ideal case, but a little shallower suggesting discretization errors are reducing the effective order of the scheme.

10.9. Boss-Bodenheimer test

The Boss-Bodenheimer test (Boss & Bodenheimer 1979) is a standard test of star formation codes designed to investigate the non-axisymmetric collapse and fragmentation of a rotating, self-gravitating gas cloud. The rotating cloud is seeded with an $m = 2$ azimuthal perturbation and therefore collapses under self-gravity and forms a bar-like structure. At the ends of the bar, dense condensations are formed.

Table 2. Initial conditions for the figure-eight 3-body problem (Chencier & Montgomery 2000; Cols. 1 to 6) and the Burrau 3-body problem (Burrau 1913; Cols. 7 to 12).

FIGURE-EIGHT	ID	m	x	y	v_x	v_y	BURRAU	ID	m	x	y	v_x	v_y
1		1.0	0.97000436	−0.2430875	0.466203685	0.43236573	1		3.0	1.0	3.0	0.0	0.0
2		1.0	−0.97000436	0.2430875	0.466203685	0.43236573	2		4.0	−2.0	−1.0	0.0	0.0
3		1.0	0.0	0.0	−0.93240737	−0.86473146	3		5.0	1.0	−1.0	0.0	0.0

Notes. In both problems, the centre of mass is at the origin, the net linear and angular momenta are zero, and dimensionless units are used, such that $G = 1$.

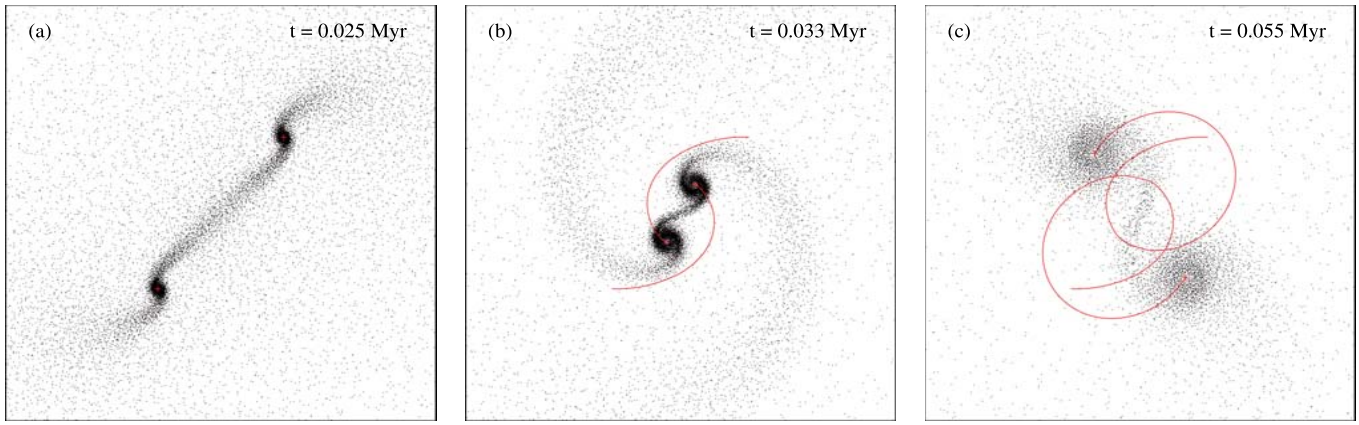

Fig. 13. SPH and sink particle plots of Boss-Bodenheimer test at times **a)** $t = 0.025$ Myr, **b)** $t = 0.033$ Myr and **c)** $t = 0.055$ Myr. SPH particles are represented by black dots (only one in every three plotted for clarity) and the position and motion of the sink particles are represented by the red lines. The first tile ($t = 0.025$ Myr) shows the particle distribution just after the formation of the two sinks in the condensations that form either end of the bar. The subsequent times shows the motion of the sink particles as they move with the gas and the small disks that form around each sink. All figures show the region $-0.005 < x < 0.005$, $-0.005 < y < 0.005$.

The initial conditions are set up as follows. A relaxed (i.e. glass-like) uniform density sphere (see Sect. 10.1, for details) is rescaled to produce the correct total mass, $M = 1\,M_\odot$, radius, $R = 3.2 \times 10^{16}$ cm, and density $\rho_0 = 1.44 \times 10^{-17}$ g cm^{-3}. We then add a sinusoidal, azimuthal density perturbation of the form

$$\rho = \rho_0 \left[1 + A \sin(m\phi)\right] \tag{68}$$

where ϕ is the azimuthal angle about the z-axis, $A = 0.5$ is the magnitude of the perturbation, and $m = 2$ is the order of the azimuthal perturbation. The density perturbation is achieved by altering the particle positions rather than changing the masses of the particles. The cloud is initially set in solid-body rotation with an angular velocity of $\Omega = 1.6 \times 10^{-12}$ rad s^{-1}. In our simulation, we use a barotropic EOS (Eq. (26) with $T_0 = 10$ K and $\rho_{\rm CRIT} = 10^{-14}$ g cm^{-3}) in order to set a minimum scale for fragmentation of the cloud. We use sink particles with a sink formation density of $\rho_{\rm SINK} = 2 \times 10^{-12}$ g cm^{-3} and sink radius $r_{\rm SINK} = 2\,h_{\rm FORM}$ where $h_{\rm FORM}$ is the smoothing length of the SPH particle that triggers sink formation. The freefall collapse timescale of the original unperturbed cloud is $t_{\rm FF} = 17.4$ yr. We use 50 000 SPH particles in the original cloud in order to adequately resolve gravitational fragmentation with our choice of EOS (Bate & Burkert 1997; Hubber et al. 2006). The simulation is run until a time of $t = 100$ kyr.

The gas initially collapses under self-gravity to form a thin, dense "bar" with two denser condensations at either end. The barotropic EOS (along with the relatively low resolution of the bar) prevents the bar collapsing to high densities once its density exceeds $\rho_{\rm CRIT}$. The denser condensations at either end of the bar are able to collapse to higher densities. Eventually, the two condensations form sinks. Figure 13a shows the particle positions just after the formation of the two sinks. The gas surrounding the sinks has some angular momentum relative to the sinks (from the original rotational field of the cloud) and assembles into two small disks which are connected together by the bar. Subsequently, the two sinks follow eccentric orbits with a series of close approaches, during which the increased compression loads more mass into the disks, from both the surrounding gas and the bar. This leads to a period of rapid accretion, followed by a relatively quiet period as the sinks move towards apastron and the accretion rate drops off. We note that in the presence of a reservoir of gas that constantly feeds accretion, the orbital properties of the system change continuously until the gas supply becomes negligible.

10.10. 3-body tests

Since the N-body integrator in SEREN is intended to follow small-N systems, it is appropriate to perform 3-body tests for which accurate solutions are known (rather then large-N tests to which only statistical constraints can be applied). We limit ourselves to two such tests.

The first test is the figure-eight 3-body problem defined by Chenciner & Montgomery (2000). The initial conditions for this test are summarised in Table 2 (left side). With these initial conditions, all three particles follow the same figure-eight trajectory, with period $\mathcal{P} = 6.32591398$. We have evolved this system using SEREN's Hermite N-body integrator with a timestep multiplier of

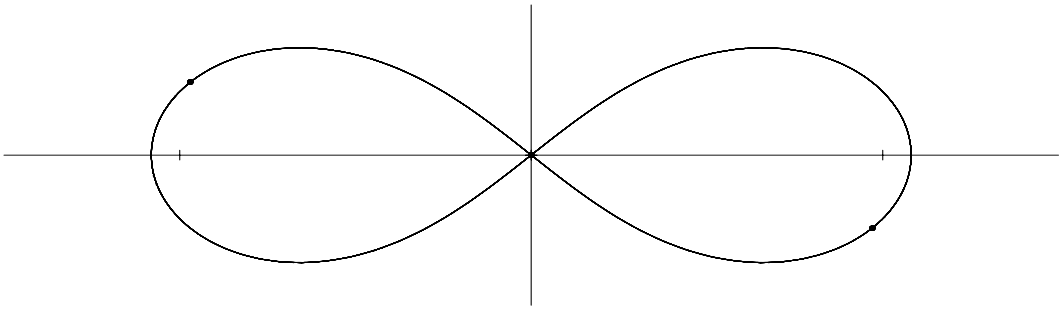

Fig. 14. Tracks for the first 20 periods of the figure-eight 3-body problem. The positions of the stars are plotted every period, with solid black dots.

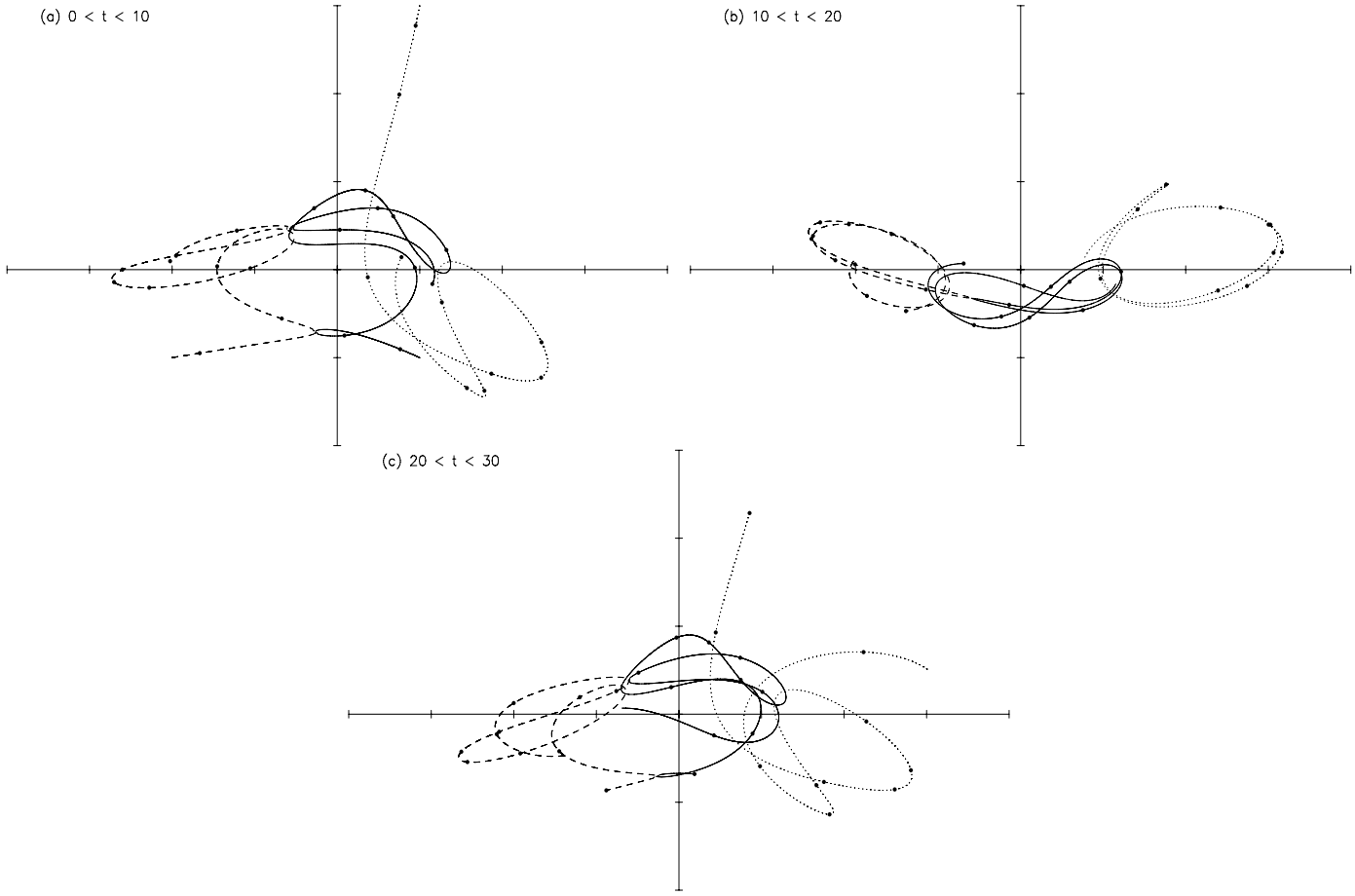

Fig. 15. Tracks for the Burrau problem in the time intervals **a)** $0 < t < 10$, **b)** $10 < t < 20$, and **c)** $20 < t < 30$. The dotted lines track star 1, the dashed lines star 2, and the solid lines star 3. Each track includes solid dots at intervals of one time unit (i.e. at $t = 1, 2, 3,$ etc.). These tracks should be compared with those presented by Szebehely & Peters (1967, their Figs. 2–4).

$\gamma = 0.05$, without the trajectory being corrupted. Figure 14 shows the trajectory, and the positions of the three stars at $t = \mathcal{NP}$ for $\mathcal{N} = 0, 1, 2, \ldots, 20$, demonstrating that the stars return to the same positions every period. After 100 orbits, energy is conserved to better than one part in 10^6, and the errors in the net linear and angular momenta are of order machine rounding error. The Hermite integrator therefore appears to be very stable.

The second test is the 3-body problem devised by Burrau (1913), in which three particles are placed at the vertices of a right-angled triangle with sides 5, 4 and 3, and each particle has a mass equal to the length of the side opposite it. The initial conditions for this test are summarised in Table 2 (right side). The subsequent evolution involves close encounters (separations $|\Delta \boldsymbol{r}_{ij}| \lesssim 10^{-3}$), and is therefore highly chaotic. The Burrau problem was first integrated numerically, to the point where one star is ejected permanently, by Szebehely & Peters (1967), using the two-dimensional Levi-Civita (1904) regularisation method. We have evolved this system up to $t = 70$, using SEREN's Hermite integrator with a low timestep multiplier of $\gamma = 0.02$ and a smoothing length of $h = 10^{-4}$. Energy is conserved to one part in 10^7, and the errors in the net linear and angular momenta are of order machine rounding error. In Figs. 15 and 16 we plot orbital tracks for the same time intervals as Szebehely & Peters (1967) and using the same line styles. The close agreement between our tracks and those of Szebehely & Peters (1967) demonstrates the accuracy and robustness of SEREN's Hermite N-body integrator.

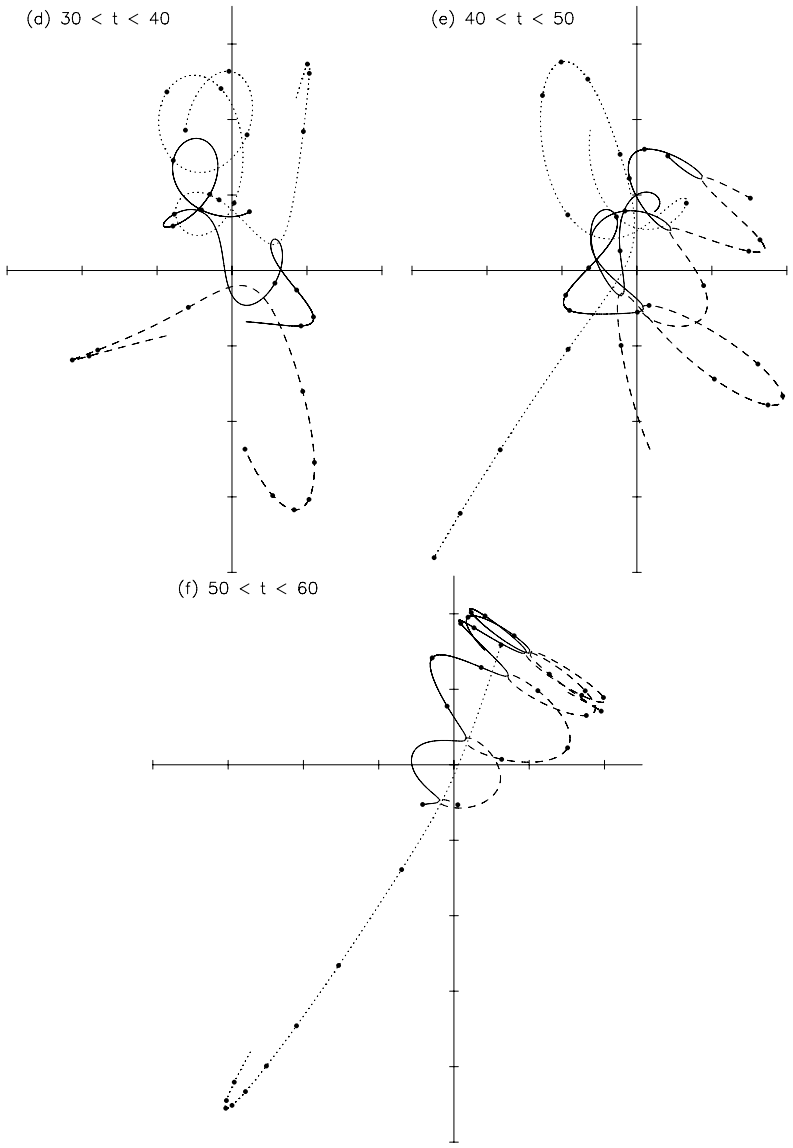

Fig. 16. As Fig. 15, but for the time intervals **d)** $30 < t < 40$, **e)** $40 < t < 50$, and **f)** $50 < t < 60$. These tracks should be compared with Figs. 5–7 in Szebehely & Peters (1967).

11. Memory and cache optimisations

Here we describe the features of SEREN which are designed to improve cache performance and reduce overall memory usage.

11.1. Particle re-ordering

Particle data arrays are arranged in tree-walk order, i.e. the order in which individual particles are interrogated during a tree-walk. This ensures that all particles in the same leaf cell are contiguous in memory, and particles in nearby branches are likely to be in a nearby (if not contiguous) part of the memory, i.e. they are likely to be within the same cache block. This requires that the data arrays are repeatedly re-ordered, but the computational cost of re-ordering is relatively low compared with the run time saved by optimising the cache usage. For large numbers of particles run times are more than halved. We do not use a more sophisticated space-filling curve, such as the Hilbert space-filling curve used in GADGET 2 (Springel 2005), which is optimal for distributed-memory architectures requiring large amounts of communication between nodes.

11.2. Grouping particle data

Since the run times of most SEREN simulations are dominated by the routines that compute gravitational accelerations, we group together in a single array all the data required for particle-particle gravitational interactions (i.e. position, mass, smoothing length). This optimises cache usage by ensuring that all the data required for calculating the gravitational interaction due to a particle is loaded in the same cache block, and therefore avoids thrashing the memory while loading the required variables.

11.3. Minimising memory allocation

For subroutines that compute SPH sums, we first walk the neighbour tree to obtain a potential-neighbour list. In the first instance, the code only allocates a small amount of memory to store the potential neighbour list ($N_{\rm MAX}$ elements, where $N_{\rm MAX} \ll N$), in order to reduce memory fragmentation. For example, in a 3D simulation, the expected mean number of neighbours might be $\bar{N}_{\rm NEIB} = 50$ (in the grad-h formulation, $\bar{N}_{\rm NEIB} = 32\pi\eta_{\rm SPH}^3/3$), and in this case an appropriate choice would be $N_{\rm MAX} = 200$). Then, in the rare instances where more than $N_{\rm MAX}$ potential neighbours are found (e.g. an isolated particle with a very large smoothing length), the memory is deallocated and reallocated to N elements.

12. Parallelisation

SEREN is parallelised using OpenMP, for use on shared-memory architectures (for example, symmetric multiprocessing (SMP) and non-uniform memory access (NUMA) machines). OpenMP requires that each processor can see all the data, and so there is no need for any explicit transfer of data between each processor's RAM, although there is some overhead associated with transferring data from the shared RAM to the local caches of individual processors. OpenMP works by parallelising do-loops. If the operations executed in each cycle of a loop are independent of those executed in the other cycles, this can be achieved simply by adding OpenMP directives at the beginning and end of the loop. The cycles of the loop are then farmed out to the available processors; if there are N cycles (corresponding to N particles) and $N_{\rm p}$ processors, each processor receives $N_{\rm BATCH} = N/N_{\rm p}$ cycles to execute.

The scaling of a parallel code, $\mathcal{S}(N_{\rm p})$, is defined as the wall-clock time, $t(1)$, the code takes to perform a reference simulation on one processor, divided by the time, $t(N_{\rm p})$, it takes to perform the same simulation on $N_{\rm p}$ processors, i.e. $\mathcal{S}(N_{\rm p}) = t(1)/t(N_{\rm p})$. A perfectly scaling code has $\mathcal{S}(N_{\rm p}) = N_{\rm p}$, but normally scaling is less than perfect, because (i) some fraction of the code is inherently serial and cannot be parallelised (Amdahl's law); (ii) the code is not perfectly load-balanced at all times (i.e. not all processors are equally busy at all times); and (iii) there is latency, for example due to communication of data between the OpenMP master node and the slave nodes. In SEREN, these difficulties are compounded by the implementation of hierarchical block timesteps (see Sect. 6.3).

The main routines in SEREN are those that (a) construct and stock the tree, (b) determine the SPH smoothing lengths, densities and other local functions of state, (c) compute the hydrodynamic accelerations and heating terms, (d) compute the gravitational accelerations, and (e) advance the particle positions, velocities and internal energies. Of these the last four can be parallelised quite straightforwardly and effectively, but the first (tree-building) can not. In particular, the assigning of new tree cells, the construction of linked lists, and the re-ordering of particles (see Sect. 11.1) can not be parallelised efficiently, and it is these elements, along with the other smaller serial sections of code, that ultimately limit the scalability of SEREN.

Even if the operations executed in each cycle of a do-loop are independent, naïve application of OpenMP directives to the beginning and end of the do-loop will not guarantee load balancing, because the individual cycles do not necessarily entail similar amounts of computation. For example, walking the gravity tree is a much more compute-intensive operation for a particle in the densest regions of a fragmenting prestellar core than for a particle in the diffuse outer envelope of the same core. Therefore, in order to improve load balancing, the code delivers to each processor a small batch of cycles, and when the processor is finished executing these cycles it requests another batch. We find, empirically, that SEREN runs most efficiently with $N_{\rm BATCH} \sim 10^{-3}N/N_{\rm p}$.

When hierarchical block timesteps are used, SEREN maintains a list of the IDs of all the active particles (i.e. the $N_{\rm ACTIVE}$ particles whose accelerations and heating terms are being computed on the current timestep). Load balancing is then achieved by only looping over this active list, and farming out batches of size $N_{\rm BATCH} \sim 10^{-3}N_{\rm ACTIVE}/N_{\rm p}$ to the individual processors.

12.1. Scaling tests

To test the scaling of SEREN we revisit the collapse of a spherical isothermal cloud which initially is at rest with uniform density (see Sect. 10.7). We model the cloud with 10^6 particles, and follow the evolution to dimensionless time $t = 0.6$, using global timesteps. Since the cloud does not develop any complicated internal structure, this is a relatively undemanding test. Figure 17a shows the net scaling obtained on a 16-core SMP node of the Cardiff University Merlin cluster, using 1, 2, 4, 8 and 16 processors. The scaling is good up to 8 processors, but for 16 processors is starting to deteriorate ($\mathcal{S}(16) \sim 13$). This indicates that SEREN is not likely to be able to exploit SMP machines with 100+ cores. Figure 17b shows how the main routines in SEREN scale individually. Evidently the gravity routines scale very well, almost perfectly; the SPH routines start to deteriorate at 8 processors ($\mathcal{S}_{\rm SPH}(8) \sim 7$); and the tree-building routines scale very poorly. It is the tree-building routines that limit the net scaling.

13. Future development

We are continuing to develop SEREN and add new features. Some of these features have already been implemented in our development code and will be released to the main working version of SEREN once they are fully tested and debugged. The most important of these additions are (i) an MPI-parallelised version of the code (McLeod et al., in prep.); (ii) an hybrid flux-limited diffusion and radiative cooling scheme (Forgan et al. 2009); (iii) the use of different timesteps for hydrodynamical and gravitational accelerations (Saitoh & Makino 2010); (iv) improved sink particles with feedback; (v) an integrated N-body and SPH integrator to model cluster dynamics with a live gas potential; (vi) ideal MHD using divergence cleaning and/or Euler potentials (Price & Monaghan 2004a,b, 2005; Rosswog & Price 2007a); and (vii) non-ideal MHD (Hosking & Whitworth 2004). The MPI version of SEREN will be a hybrid MPI/OpenMP code that can parallelise a group of shared memory nodes using MPI communication to link them together. This will reduce the amount of communication between nodes, which is often the bottleneck to good scalability over a large number of processors. The remaining additions to SEREN are implementations of existing algorithms. We will provide up-to-date information

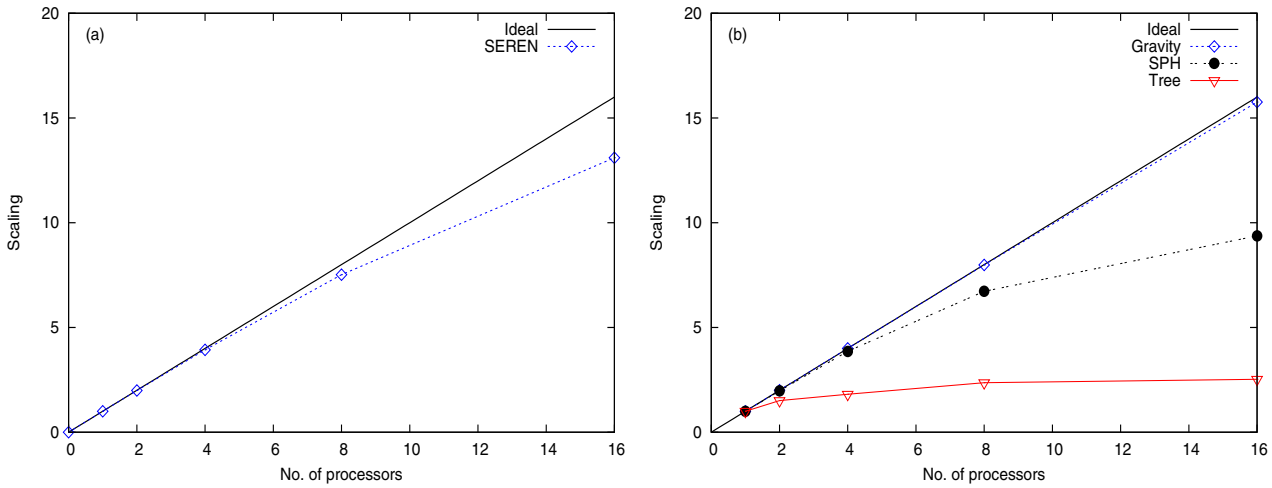

Fig. 17. **a)** Net scaling of SEREN, using OpenMP on a 16-core SMP machine, as a function of the number of processors. **b)** Scaling of the individual gravity, SPH and tree-building routines, as a function of the number of processors.

on the development status of SEREN, and any further tests we have performed, at the web address http://www.astro.group.shef.ac.uk/seren.

Acknowledgements. We would like to thank Simon Goodwin for providing the DRAGON SPH code to the authors on which some of the initial development of SEREN was based upon. We would also like to thank Sumedh Anathpindika, Thomas Bisbas, Steinar Børve, Murat Kaplan, Dimitrios Stamatellos, Jan Trulsen, Stephanie Walch & Richard Wünsch for helpful suggestions and comments during the development of the code and for donating some routines to SEREN. We also thank the anonymous referee for useful suggestions that have improved the paper. D.A.H. is funded by a Leverhulme Trust Research Project Grant (F/00 118/BJ). C.P.B. is funded by an STFC studentship. A.M. is funded by an STFC studentship and was funded as an Early-Stage Researcher by the EC-funded CONSTELLATION Marie Curie Training Network MRTN-CT-2006-035890. A.P.W. gratefully acknowledges the support of the STFC rolling grant PP/E000967/1, and the CONSTELLATION network. We thank Daniel Price for the use SPLASH (Price 2007a) for creating some images.

Appendix A: Kernel functions

A.1. M4 cubic spline kernel

The M4 cubic spline kernel (Monaghan & Lattanzio 1985) is used in many implementations of SPH, due to its simple form and its compact support. The M4 kernel is a function of $s \equiv r/h$ only. For $D = 1,\ 2,$ and 3 dimensions, it takes the form

$$W(s) = \frac{\sigma_D}{h^D} \begin{cases} 1 - \frac{3}{2}s^2 + \frac{3}{4}s^3 & 0 \le s \le 1; \\ \frac{1}{4}(2-s)^3 & 1 \le s \le 2; \\ 0 & s > 2. \end{cases} \tag{A.1}$$

where $\sigma_1 = 2/3$, $\sigma_2 = 10/7\pi$, and $\sigma_3 = 1/\pi$. The first spatial derivative is

$$\frac{\mathrm{d}W}{\mathrm{d}r}(s) = -\frac{\sigma_D}{h^{D+1}} \begin{cases} 3s - \frac{9}{4}s^2 & 0 \le s \le 1; \\ \frac{3}{4}(2-s)^2 & 1 \le s \le 2; \\ 0 & s > 2. \end{cases} \tag{A.2}$$

SEREN also allows the modified derivative proposed by Thomas & Couchman (1992) to prevent the clumping instability,

$$\left.\frac{\mathrm{d}W}{\mathrm{d}r}\right|_{\mathrm{TC}}(s) = -\frac{\sigma_D}{h^{D+1}} \begin{cases} 1 & 0 \le s \le \frac{2}{3}; \\ 3s - \frac{9}{4}s^2 & \frac{2}{3} \le s \le 1; \\ \frac{3}{4}(2-s)^2 & 1 \le s \le 2; \\ 0 & s > 2. \end{cases} \tag{A.3}$$

For "grad-h" SPH, the Ω correction kernel function is given by

$$\frac{\partial W}{\partial h}(s) = \frac{\sigma_D}{h^{D+1}} \begin{cases} -D + \frac{3}{2}(D+2)\,s^2 - \frac{3}{4}(D+3)\,s^3 & 0 \le s \le 1; \\ -2\,D + 3\,(D+1)\,s - \frac{3}{2}(D+2)\,s^2 + \frac{1}{4}(D+3)\,s^3 & 1 \le s \le 2; \\ 0 & s > 2. \end{cases} \tag{A.4}$$

For kernel-softened gravity (three dimensions only), the kernel function ϕ' is

$$\phi'(s) = \frac{1}{h^2} \begin{cases} \frac{4}{3}s - \frac{6}{5}s^3 + \frac{1}{2}s^4 & 0 \le s \le 1\,; \\ \frac{8}{3}s - 3s^2 + \frac{6}{5}s^3 - \frac{1}{6}s^4 - \frac{1}{15}\frac{1}{s^2} & 1 \le s \le 2\,; \\ 1/s^2 & s > 2. \end{cases} \tag{A.5}$$

For calculating the gravitational potential, the kernel function ϕ is

$$\phi(s) = -\frac{1}{h}\begin{cases} \frac{7}{5} - \frac{2}{3}s^2 + \frac{3}{10}s^4 - \frac{1}{10}s^5 & 0 \le s \le 1\,; \\ \frac{8}{5} - \frac{4}{3}s^2 + s^3 - \frac{3}{10}s^4 + \frac{1}{30}s^5 - \frac{1}{15}\frac{1}{s} & 1 \le s \le 2\,; \\ 1/s & s > 2. \end{cases} \tag{A.6}$$

For "grad-h" gravity (Price & Monaghan 2007), the kernel function ζ is calculated using

$$\frac{\partial\phi}{\partial h}(s) = \frac{1}{h^2}\begin{cases} \frac{7}{5} - 2\,s^2 + \frac{3}{2}\,s^4 - \frac{3}{5}\,s^5 & 0 \le s \le 1\,; \\ \frac{8}{5} - 4\,s^2 + 4\,s^3 - \frac{3}{2}\,s^4 + \frac{1}{5}\,s^5 & 1 \le s \le 2\,; \\ 0 & s > 2. \end{cases} \tag{A.7}$$

A.2. Quintic spline kernel

The quintic spline kernel (Morris 1996) is a fifth-order polynomial function with compact support. It was originally presented in the form of a factorised polynomial. However, to facilitate the processes of differentiation and integration that are required to compute the other kernel functions, we expand the brackets into a simple power series:

$$W(s) = \frac{\sigma_D}{h^D}\begin{cases} 66 - 60s^2 + 30s^4 - 10s^5 & 0 \le s \le 1; \\ 51 + 75s - 210s^2 + 150s^3 - 45s^4 + 5s^5 & 1 \le s \le 2; \\ 243 - 405s + 270s^2 - 90s^3 + 15s^4 - s^5 & 2 \le s \le 3; \\ 0 & s > 3, \end{cases} \tag{A.8}$$

where $s \equiv r/h$ and $\sigma_1 = 120$, $\sigma_2 = 7/478\pi$, and $\sigma_3 = 3/359\pi$. The first spatial derivative is

$$\frac{\mathrm{d}W}{\mathrm{d}r}(s) = \frac{\sigma_D}{h^{D+1}}\begin{cases} -120s + 120s^3 - 50s^4 & 0 \le s \le 1; \\ 75 - 420s + 450s^2 - 180s^3 + 25s^4 & 1 \le s \le 2; \\ -405 + 540s - 270s^2 + 60s^3 - 5s^4 & 2 \le s \le 3; \\ 0 & s > 3. \end{cases} \tag{A.9}$$

For "grad-h" SPH,

$$\frac{\partial W}{\partial h}(s) = -\frac{\sigma_D}{h^{D+1}}\begin{cases} 66D - 60(D+2)s^2 + 30(D+4)s^4 - 10(D+5)s^5 & 0 \le s \le 1; \\ 51D + 75(D+1)s - 210(D+2)s^2 + 150(D+3)s^3 - 45(D+4)s^4 + 5(D+5)s^5 & 1 \le s \le 2; \\ 243D - 405(D+1)s + 270(D+2)s^2 - 90(D+3)s^3 + 15(D+4)s^4 - (D+5)s^5 & 2 \le s \le 3; \\ 0 & s > 3. \end{cases} \tag{A.10}$$

For kernel-softened gravity (three dimensions only), the kernel function ϕ' is

$$\phi'(s) = \frac{1}{h^2}\begin{cases} \frac{12}{359}\left(22s - 12s^3 + \frac{30}{7}s^5 - \frac{5}{4}s^6\right) & 0 \le s \le 1\,; \\ \frac{12}{359}\left(17s + \frac{75}{4}s^2 - 42s^3 + 25s^4 - \frac{45}{7}s^5 + \frac{5}{8}s^6 + \frac{5}{56}\frac{1}{s^2}\right) & 1 \le s \le 2\,; \\ \frac{12}{359}\left(81s - \frac{405}{4}s^2 + 54s^3 - 15s^4 + \frac{15}{7}s^5 - \frac{1}{8}s^6 + \frac{507}{56}\frac{1}{s^2}\right) & 2 \le s \le 3\,; \\ 1/s^2 & s > 3\,. \end{cases} \tag{A.11}$$

The gravitational potential kernel ϕ is (three dimensions only)

$$\phi(s) = -\frac{1}{h}\begin{cases} \frac{12}{359}\left(\frac{478}{14} - 11s^2 + 3s^4 - \frac{5}{7}s^6 + \frac{5}{28}s^7\right) & 0 \le s \le 1\,; \\ \frac{12}{359}\left(\frac{473}{14} - \frac{17}{2}s^2 - \frac{25}{4}s^3 + \frac{21}{2}s^4 - 5s^5 + \frac{15}{14}s^6 - \frac{5}{56}s^7 + \frac{5}{56}\frac{1}{s}\right) & 1 \le s \le 2\,; \\ \frac{12}{359}\left(\frac{729}{14} - \frac{81}{2}s^2 + \frac{135}{4}s^3 - \frac{27}{2}s^4 + 3s^5 - \frac{5}{14}s^6 + \frac{1}{56}s^7 - \frac{507}{56}\frac{1}{s}\right) & 1 \le s \le 2\,; \\ 1/s & s > 3\,. \end{cases} \tag{A.12}$$

For "grad-h" gravity (three dimensions only), the kernel function ζ is calculated using

$$\frac{\partial\phi}{\partial h}(s) = \frac{1}{h}\begin{cases} \frac{12}{359}\left(\frac{478}{14} - 33s^2 + 15s^4 - 5s^6 + \frac{10}{7}s^7\right) & 0 \le s \le 1\,; \\ \frac{12}{359}\left(\frac{473}{14} - \frac{51}{2}s^2 - 25s^3 + \frac{105}{2}s^4 - 30s^5 + \frac{15}{2}s^6 - \frac{5}{7}s^7\right) & 1 \le s \le 2\,; \\ \frac{12}{359}\left(\frac{729}{14} - \frac{243}{2}s^2 + 135s^3 - \frac{135}{2}s^4 + 18s^5 - \frac{5}{2}s^6 + \frac{1}{7}s^7\right) & 1 \le s \le 2\,; \\ 0 & s > 3\,. \end{cases} \tag{A.13}$$

Appendix B: Multipole moments

When calculating gravitational accelerations, using the Barnes-Hut gravity tree, SEREN calculate the contribution from a cell up to octupole order, if requested. The multipole moments of each cell are computed relative to the centre of mass of the cell; this means that the dipole term is always zero. The components of the quadrupole moment tensor, Q, for a leaf cell, c, are given by

$$Q_{ab,c} = \sum_i m_i\left(3\,x_{a,i}x_{b,i} - r_i^2\delta_{ab}\right), \tag{B.1}$$

where the summation is over all the particles i in the leaf cell. If the cell is not a leaf cell, the quadrupole moment tensor is given by

$$Q_{ab,c} = \sum_d m_d \left(3\, x_{a,d} x_{b,d} - r_d^2 \delta_{ab}\right) + \sum_d Q_{ab,d}, \tag{B.2}$$

where the summation is over all the daughter cells d. The octupole moment tensor, S, for a leaf cell, c, is given by

$$S_{ab,c} = \sum_i m_i \left[5\,(3 - 2\delta_{ab})\, x_{a,i}^2 - 3 r_i^2\right] x_{b,i}, \tag{B.3}$$

$$S_{123,c} = 15 \sum_i m_i x_{1,i} x_{2,i} x_{3,i}; \tag{B.4}$$

and for a non-leaf cell by

$$S_{ab,c} = \sum_d m_i \left[5\,(3 - 2\delta_{ab})\, x_{a,d}^2 - 3 r_d^2\right] x_{b,d} + \sum_d \left[5\,(1 - \delta_{ab})\, x_{a,d} Q_{ab,d} + \tfrac{5}{2} x_{b,d} Q_{aa,d} - x_{l,d} Q_{bl,d} + S_{ab,d}\right], \tag{B.5}$$

$$S_{123,c} = 15 \sum_d m_i x_{1,d} x_{2,d} x_{3,d} + \sum_d \left[\tfrac{5}{3}\left(x_{1,d} Q_{23,d} + x_{2,d} Q_{31,d} + x_{3,d} Q_{12,d}\right) + S_{123,d}\right]. \tag{B.6}$$

References

Aarseth, S. J. 2001, NewA, 6, 277
Aarseth, S. J. 2003, Gravitational *N*-body Simulations: Tools and Algorithms (Cambridge University Press)
Abel, T., Bryan, G. I., & Norman, M. L., 2002, Science, 295, 93
Agertz, O., Moore, B., Stadel, J., et al. 2007, MNRAS, 380, 963
Balsara, D. S. 1995, JCoPh, 121, 357
Barnes, J., & Hut, P. 1986, Nature, 324, 446
Bate, M. R., & Burkert, A. 1997, MNRAS, 288, 1060
Bate, M. R., Bonnell, I. A., & Price, N. M. 1995, MNRAS, 277, 362
Benz, W. 1990, Numerical Modelling of Nonlinear Stellar Pulsations: Problems and Prospects (Kluwer Academic Publishers), ed. Buchler, J. R., 269
Bisbas, T. G., Wünsch, R., Whitworth, A. P., & Hubber, D. A. 2009, A&A, 497, 649
Boss, A. P., & Bodenheimer, P. 1979, ApJ, 234, 289
Burrau, C. 1913, AN, 195, 113
Cartwright, A., & Stamatellos, D. 2010, A&A, 516, A99
Cha, S. H., & Whitworth, A. 2003, MNRAS, 340, 73
Chandrasekhar, S. 1939, An Introduction to the Study of Stellar Structure (New York: Dover Publs. Inc.;)
Chenciner, A., & Montgomery, R. 2000, AnMat, 152, 881
Delgado Donate, E. J., Clarke, C. J., & Bate, M. R. 2003, MNRAS, 342, 1926
Forgan, D., Rice, K., Stamatellos, D., & Whitworth, A. P. 2009, MNRAS, 394, 882
Fryxell, B., Olson, K., Ricker, P., et al. 2000, ApJS, 131, 273
Gingold, R. A., & Monaghan, J. J. 1977, MNRAS, 181, 375
Hernquist, L. 1987, ApJS, 64, 715
Hernquist, L., & Katz, N. 1989, ApJS, 70, 419
Hernquist, L., Bouchet, F. R., & Suto, Y. 1991, ApJS, 75, 231
Hosking, J. G., & Whitworth, A. P. 2004, MNRAS, 347, 994
Hubber, D. A., Goodwin, S. P., & Whitworth, A. P. 2006, 450, 881
Inutsuka, S. 2002, JCoPh, 179, 238
Klessen, R. S. 1997, MNRAS, 292, 11
Klessen, R. S., Heitsch, F., & Mac Low, M.-M. 2000, ApJ, 535, 887
Levi-Civita, T. 1904, Ann. Mat. Ser., 9, 1
Lucy, L. 1977, AJ, 82, 1013
Martin, G. E. 1998, Geometric constructions (Springer-Verlag)
Makino, J. 1991, ApJ, 369, 200
Makino, J., & Aarseth, S. J. 1992, PASJ, 44, 141
Makino, J., Fukushige, T., Koga, M., & Namura, K. 2003, PASJ, 55, 1163
McMillan, S. L. W., & Aarseth, S. J. 1993, AJ, 414, 200
Merlin, E., Buonomo, U., Grassi, T., Piovan, L., & Chiosi, C. 2010, A&A, 513, 36
Monaghan, J. J. 1992, ARA&A, 30, 543
Monaghan, J. J. 1997, JCoPh, 136, 298
Monaghan, J. J. 2002, MNRAS, 335, 843
Monaghan, J. J. 2005, RPPh, 68, 1703
Monaghan, J. J. 2006, MNRAS, 365, 199
Monaghan, J. J., & Gingold, R. A. 1983, J. Comp. Phys, 52, 374
Monaghan, J. J., & Lattanzio, J. C. 1985, A&A, 149, 135
Morris, J. P. 1996, PhD Thesis – Analysis of Smoothed Particle Hydrodynamics with Applications, Monash University
Morris, J. P., & Monaghan, J. J. 1997, JCoPh, 136, 41
Nelson, R. P., & Papaloizou, J. C. B. 1994, MNRAS, 270, 1
Nitadori, K., & Makino, J. 2008, NewA, 13, 498
Pfalzner, S., & Gibbon, S. 1996, Many-body tree methods in Physics (Cambridge University Press)
Portegies Zwart, S. F., McMillan, S. L.W, Hut, P., & Makino, J. 2001, MNRAS, 321, 199
Price, D. J. 2007, PASA, 24, 159
Price, D. J. 2008, JCoPh, 227, 10040
Price, D. J., & Monaghan, J. J. 2004a, MNRAS, 348, 123
Price, D. J., & Monaghan, J. J. 2004b, MNRAS, 348, 139

Price, D. J., & Monaghan, J. J. 2005, MNRAS, 364, 384
Price, D. J., & Monaghan, J. J. 2007, MNRAS, 374, 1347
Read, J. I., Hayfield, T., & Agertz, O. 2010, MNRAS, 405, 1513
Riley, K. F., Hobson, M. P., & Bence, S. J. 1998, Mathematical methods for physics and engineering (Cambridge University Press)
Rosswog, S., & Price, D. J. 2007, MNRAS, 379, 915
Rosswog, S., Davies, M.B., Thielemann, F.-K., & Piran, T. 2000, A&A, 360, 171
Saigo, K., & Tomisaka, K. 2006, ApJ, 645, 381
Saitoh, T. R., & Makino, J. 2009, ApJ, 697, 99
Saitoh, T. R., & Makino, J. 2010, PASJ, 62, 301
Sedov, L. I. 1959, Similarity and Dimensional Methods in Mechanics (New York: Academic Press)
Sod, G. A. 1978, JCoPh., 27, 1
Springel, V. 2005, MNRAS, 364, 1105
Springel, V. 2010, MNRAS, 401, 791
Springel, V., & Hernquist L. 2002, MNRAS, 333, 649
Springel, V., White, S. D. M., Jenkins, A., et al. 2005, Nature 435, 629
Springel, V., Yoshida, N., & White, S. D. M. 2001, NewA, 6, 79
Stamatellos, D., Whitworth, A. P., Bisbas, T., & Goodwin, S. P. 2007, MNRAS, 475, 37
Stiefel, E. L., & Scheifele, G. 1971, Linear and regular celestial mechanics (Springer-Verlag)
Stone, J. M., & Norman, M. L. 1992, AJS, 80, 753
Szebehely, V., & Peters, C. F. 1967, AJ, 72, 876
Teyssier, R. 2002, A&A, 385, 337
Thomas, P. A., & Couchman, H. M. P. 1992, MNRAS, 257, 11
Truelove, J. K., Klein R. I., McKee, C. F., et al. 1998, ApJ, 495, 821
Van Albada, T. S. 1968, BAN, 19, 479
Wadsley, J. W., Stadel, J., & Quinn, T. 2004, NewA, 8, 137
Wetzstein, M., Nelson, A. F., Naab, T., & Burkert, A. 2009, ApJS, 184, 298