Mon. Not. R. Astron. Soc. **000**, 000–000 (0000) Printed 14 September 2016 (MN LaTeX style file v2.2)

HERMES: a hybrid integrator for simulating close encounters and planetesimal migration

Ari Silburt[1,2], Hanno Rein[2,1] & Dan Tamayo[2,3,4]
[1] *Department of Astronomy and Astrophysics, University of Toronto, Toronto, Ontario, M5S 3H4, Canada*
[2] *Department of Physical and Environmental Sciences, University of Toronto at Scarborough, Toronto, Ontario M1C 1A4, Canada*
[3] *Canadian Institute for Theoretical Astrophysics, 60 St. George St, University of Toronto, Toronto, Ontario M5S 3H8, Canada*
[4] *Centre for Planetary Sciences Fellow*

Draft version: 14 September 2016

ABSTRACT
We present HERMES, a new hybrid integration scheme for long-term simulations of planetary systems undergoing close encounters or planetesimal-driven migration. Particles are integrated using WHFast, a fast and accurate symplectic integrator, unless a close encounter occurs. During a close encounter, a subset of particles is integrated with the high-order integrator IAS15, while the rest of the particles continue to be integrated with WHFast. We created an adaptive routine for optimizing the close encounter boundary to help maintain accuracy whilst close encounters are occurring.

The switching between integrators leads to a finite energy error. HERMES takes a more direct approach when switching between integrators than previous schemes in the literature, allowing us to analytically estimate the numerical error of our algorithm. Since WHFast is symplectic, IAS15 is accurate to machine precision and both of them are unbiased, the energy error grows sub-linearly with time under the assumption that either impact parameters are randomly distributed or close encounters are rare.

We find that HERMES provides a good balance between speed and accuracy, neither achieved by the individual symplectic or non-symplectic integrators alone. In this paper, we describe the details of implementation, accuracy and performance, as well as its incorporation within the larger framework of the N-body package REBOUND.

1 INTRODUCTION

Over the last 25 years scientists have made considerable progress integrating N gravitationally interacting particles (an N-body system) using computational techniques. The most widely used integrator today for solving Solar System type problems is the Wisdom-Holman integrator (Wisdom & Holman 1991, hereafter WH), which decomposes the system's Hamiltonian, H, into a Keplerian and an interaction component, H_K and H_I. Symplectic integrators which split the Hamiltonian in this way are known as mixed-variable symplectic integrators (Wisdom & Holman 1991; Saha & Tremaine 1992). The system is then evolved in a second-order leapfrog manner, taking the form of $K(\mathrm{dt}/2)I(\mathrm{dt})K(\mathrm{dt}/2)$, where K represents evolution under H_K, I represents evolution under H_I, and dt is the timestep. Although higher-order algorithms are possible (e.g. Yoshida 1990), the second-order WH method is optimal since the increased accuracy of higher-order methods comes at the cost of additional calculation. The evolution under the interaction Hamiltonian is trivial to solve exactly in Cartesian coordinates, whereas the evolution under the Keplerian Hamiltonian is easy to solve exactly using orbital elements. This algorithm therefore converts between the two coordinate systems each timestep.

Since the WH scheme breaks the evolution into operators that both derive from Hamiltonians, the algorithm is symplectic (for a review on symplectic algorithms see Yoshida (1993)). This implies that the numerical solution conserves quantities closely related to the integrals of motion, such as the total energy. The relative energy error scales as $O(\epsilon dt^2)$ if the magnitude of H_I remains $O(\epsilon)$ smaller than H_K, where $\epsilon \ll 1$ (Saha & Tremaine 1994). For distant particles in non-overlapping orbits ϵ is typically much less than unity. This is one motivation for splitting H into H_K and H_I, as it allows for longer timesteps (and thus shorter integration times) than conventional integration schemes. However, during close encounters, H_I becomes comparable to or larger than H_K causing ϵ, and thus the energy error, to grow substantially. Therefore, despite their brief duration, close encounters typically dictate an unacceptably short timestep for the entire simulation. Note that it is not possible to dynamically change the timestep in the standard WH integrator as it would break time symmetry and thus symplecticity[1].

For very high accuracy integrations (with relative errors of order the machine precision), non-symplectic integrators are as good and as fast as or faster than symplectic integrators (Rein & Spiegel 2015). But in most integrations, medium to low accuracy is enough to capture the qualitative evolution of a system. In such a case a symplectic integrator provides an advantage as the timestep can be large while keeping the numerical errors bound. This advantage of symplectic integrators, together with the common need to accurately resolve close encounters motivates the development of hybrid integrators that can make use of both symplectic and non-symplectic integrators.

Several hybrid integrators that make use of a (modified) WH integrator have been developed. The two most popular ones are `SyMBA` (Duncan et al. 1998) and the hybrid integrator from the `MERCURY` package (Chambers 1999, hereafter referred to as `MERCURY`).

`SyMBA` decomposes the interaction potential into a series of shells around each body and uses progressively smaller timesteps for each shell to increase time resolution. If the particles are well separated, there is only one contributing shell to the interaction term and the integrator is effectively WH. During a close encounter the inner shells contribute to the integration using smaller timesteps to resolve the encounter.

`MERCURY` handles close encounters by using a smooth changeover function to transfer large terms from H_I to H_K. When particles are distant, interactive forces between particles are small and evaluated during H_I. When particles are close, interactive forces become large and are transferred to H_K, thus keeping H_I small. This makes H_K a three body problem (central body plus two particles undergoing a close encounter) which cannot be solved analytically but is straightforward to integrate numerically to high precision using a Bulirsch-Stoer routine (Press et al. 1988).

In this paper we present a new integration method, `HERMES`, which borrows ideas from the integrators mentioned above, but takes a more direct approach to handling close encounters. It combines two existing integrators, `WHFast` (Rein & Tamayo 2015) which is a fast and unbiased implementation of the WH method, and the high-order `IAS15` integrator (Rein & Spiegel 2015). `HERMES` has been seamlessly incorporated into `REBOUND` (Rein & Liu 2012), adding further flexibility to the modular N-body package.

The outline for the paper is as follows: Section 2 describes the algorithm for `HERMES`, Section 3 characterizes error, Section 4 shows standard tests of `HERMES` as well as a comparison to `SyMBA` and `MERCURY`, and we conclude in Section 5.

[1] If one can make the timestep choice independent of the current state, for example by using a predefined sequence of timesteps, then the integrator remains symplectic.

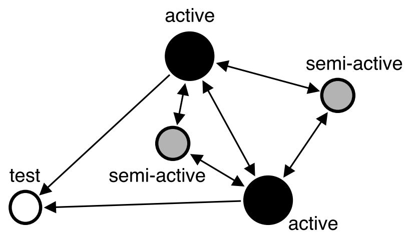

Figure 1. A diagram illustrating how different particle types (active, semi-active, test) affect each other. Arrows indicate directions of gravitational influence.

2 METHODS

2.1 Particle Classification

First we define the three different types of particles handled by `HERMES`: active particles, semi-active particles and test particles. Active particles can gravitationally affect all other types of particles, and are typically stars or planets. Semi-active particles can affect active particles only (not other semi-active particles), and are typically asteroids, planetesimals, and other smaller objects. Test particles are only affected by active particles and cannot affect any other particle, and are typically dust grains, rocks, small asteroids or spacecrafts. Figure 1 summarizes these interactions, where arrows represent directions of gravitational influence.

2.2 Heliocentric version of `WHFast`

The original `WHFast` algorithm described in Rein & Tamayo (2015) was implemented in Jacobi coordinates. Jacobi coordinates lead to a better precision compared to heliocentric coordinates if orbits are well separated and do not cross each other. If close encounter occur, then heliocentric coordinates can help improve the integrator's accuracy. For that reason we implemented a heliocentric version of `WHFast` in `REBOUND`. We call it `WHFastHelio`. We use it as the symplectic integrator in `HERMES` but note that `WHFastHelio` can also be used by itself. We choose the specific splitting of the Hamiltonian by Duncan et al. (1998) and Chambers (1999). For a discussion on the different splittings, their advantages and disadvantages, see Wisdom (2006). For the remainder of this paper we will refer to `WHFastHelio` simply as `WHFast` as both integrators use the same Kepler solver.

2.3 Algorithm

The `HERMES` integrator is composed of two parts, a *global* simulation which contains all particles, and a *mini* simulation which contains all active particles plus any semi-active or test particles involved in a close encounter. The global simulation is integrated using `WHFast`, while the mini simulation is integrated using `IAS15`. We first outline the overall algorithm for one timestep[2] of length dt and then describe the individual steps in more detail.

[2] We refer to the timestep that `WHFast` takes as dt. The `IAS15` integrator chooses its own timestep which is typically smaller than dt.

To evolve the system for a single timestep HERMES performs the following steps:

(i) **Check for close encounters.** If any particle (active, semi-active, or test) has a close encounter with an active particle, then copy those particles involved in the close encounter as well as all active particles to the mini simulation.

(ii) **Integrate the global simulation** using the WHFast integrator for one timestep, dt.

(iii) **Integrate the mini simulation** using the IAS15 integrator. The mini simulation only needs to be computed if a close encounter is currently underway and there are particles in the mini simulation.

(iv) **Update the particles in the global simulation** if the mini simulation was active this timestep.

Although the algorithm is simple to write down in the above form, there are several caveats to point out. For all particles excluding the central body, we define the parameter $f_{\rm H}$, which we dub the Hill Switch Factor. A spherical shell is constructed around each body from the object's Hill radius r_H times $f_{\rm H}$, and if the shells of any two particles overlap it is deemed a close encounter. Since all semi-active and test particles are invisible to each other they cannot be involved in close encounters with one another. Only particles that gravitationally interact with each other can participate in close encounters (see Fig. 1). Whenever there is at least one close-encounter the mini simulation is integrated, if no close-encounters occur, the mini simulation is not active and the integrator defaults to WHFast. Since the central object has no Hill sphere this motivates us to define the Solar Switch Factor, $f_\odot$, which only applies to the central body. Like $f_{\rm H}$ it also defines a spherical shell except is in units of the star's physical radius instead of Hill radii.

During a close encounter, WHFast still integrates all particles (including those involved in the close encounter) leading to momentarily large errors for the particles involved in the close encounter. One might expect that this poses a real problem for the accuracy, but that is not the case, since all particles involved in the close encounter plus all active particles are overwritten at the end of the timestep using the accurate results from the mini simulation.

Unlike the global simulation, the mini simulation may take many timesteps to get from t to $t+dt$. The length of the timestep in the mini simulation is automatically determined by the IAS15 integrator. During each sub-timestep the mini simulation also checks for physical collisions between overlapping particles.

As an example of how the mini and global simulations integrate through time, consider a 2 planet, 2 planetesimal system. The planets are active particles and the planetesimals are semi-active particles. Figure 2 shows the distance of the planetesimals and planet 2 from planet 1 as a function of time. A point is plotted after every timestep in both the global and mini simulation. After 0.2 years, planetesimal 1 (a semi-active body) has a close encounter with planet 1 (an active body). At that time, the mini simulation is turned on and planetesimal 1, planet 1 the central star (not shown) and planet 2 are added to the mini simulation and integrated until 0.75 years, at which point the close encounter between planetesimal 1 and planet 1 is complete. Planetesimal 2 on the other hand continues to be solely integrated by the global simulation (using WHFast) throughout the close encounter. By comparing the outputs of planetesimal 2 and the other particles in Fig. 2, one can see that the mini simulation takes numerous sub-timesteps compared to the global simulation. Since IAS15 is an adaptive method, it automatically chooses the appropriate timestep to resolve the close encounter with machine precision accuracy.

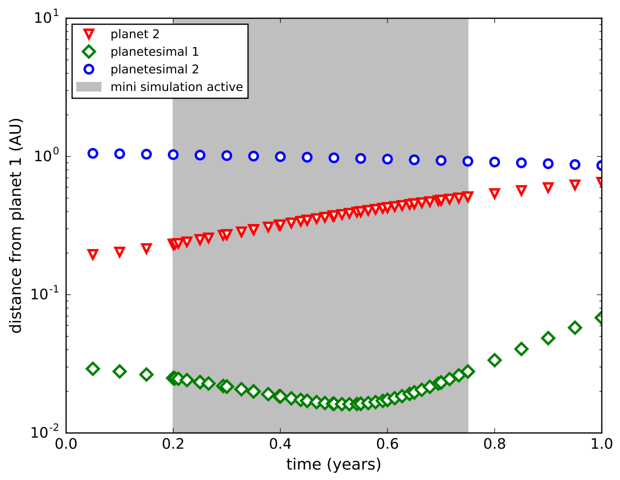

Figure 2. A short simulation displaying the HERMES integrator for a 2 planet, 2 planetesimal system orbiting a central star. When active, the mini simulation takes many sub-timesteps for each dt and integrates planets 1, 2 and planetesimal 1 during the close encounter between planet 1 and planetesimal 1.

We conclude by discussing the speed of the algorithm. If no particles are integrated with IAS15, then the speed is effectively that of WHFast with a small overhead due to collision checks. If all particles are integrated with IAS15, then the speed is that of a simulation running only IAS15, again with a small and in general negligible overhead due to collision checks. Consider a typical simulation of multiple active particles undergoing planetesimal migration from a large number of semi-active particles with a reasonable $f_{\rm H}$ value. As the number of semi-active particles is increased, the ratio of the number of particles in the mini simulation, $N_{\rm mini}$, to the number of particles in the global simulation, $N_{\rm global}$, approaches a constant. In this limit the elapsed simulation time is linearly proportional to the number of semi-active particles in the simulation.

2.4 Perturbative Forces in the Mini Simulation

One must carefully treat the forces perturbing the motions of active particles in the mini simulation. In the global simulation, active particles receive perturbative kicks from all semi-active particles, and it is important to reproduce these forces in the mini simulation, which only evolve a subset of particles. To further complicate this issue, the mini simulation takes numerous sub-timesteps for each global timestep. In HERMES, we linearly interpolate the forces of all semi-active particles absent from the mini simulation using the initial and final values from the global simulation. We find that interpolating the forces (rather than positions) leads to a significant speed gain without compromising noticeable accuracy.

One could argue that this process should be iterated – the active particles will arrive at slightly different (and more accurate) final positions when integrated with the mini simulation, and thus the perturbative forces they would have induced on the semi-active particles in the global simulation would be slightly different too. Thus if one wanted to improve the accuracy of the algorithm an iterative process could be constructed where the global and mini simulations take turns integrating for a timestep dt making use of the updated positions from the previous iteration.

However, as long as the semi-active particles have a much smaller mass than the active particles, we have found that this iterative process is unnecessary, allowing us to reduce both the computation time and algorithmic complexity. We note that in our current non-iterative scheme, interpolating the forces between pairs of *active* particles introduces non-negligible numerical errors. For that reason, all active particles are automatically added to the mini simulation during any close encounter, even if they are not involved in the close encounter themselves.

2.5 Adaptive $f_{\rm H}$ Algorithm

$f_{\rm H}$ and dt are the most important parameters to consider when simulating a system with HERMES. For a system free of close encounters, dt alone determines the precision of the algorithm. However during close encounters $f_{\rm H}$ and dt together determine the algorithm's precision (see Section 3). Specifically, if a particle moves a distance $\sim f_{\rm H} r_H$ per timestep dt the algorithm could miss a close encounter, introducing large errors into the simulation. In addition, an initial choice of $f_{\rm H}$ and dt can become non-optimal if a system evolves significantly from its initial state.

To aid the user in making the correct parameter choices, we have developed a simple algorithm that, given a timestep dt, conservatively estimates the smallest value of $f_{\rm H}$ under the condition that no close encounter is missed. Although $f_{\rm H}$ and dt both determine the precision of HERMES during a close encounter, we only optimize $f_{\rm H}$ since constantly changing dt would result in non-negligible numerical errors for a symplectic integrator like WHFast. We calculate the optimal $f_{\rm H}$ each iteration, ensuring that $f_{\rm H}$ adapts to an evolving system and guarantees that close encounters are continuously resolved. The user is therefore only required to set the timestep dt for a standard integration ($f_\odot$ is set to a default value serving most purposes). The full algorithm works as follows.

For each body, we ignore the inclination, and marginalize over the phase and longitude of periapsis from 0 to 2π. As illustrated in Figure 3, this smears out the orbit into a ring in the reference plane. Each ring has a maximum and minimum distance from the central object, $r_{\rm min}$ and $r_{\rm max}$. We then check if any two interacting particles can possibly have a close encounter by comparing their $r_{\rm min}$ and $r_{\rm max}$ values. If an intersection of two rings occurs, say for particles i and j, we calculate their maximum relative velocity $\Delta v_{ij,\rm max}$ in the overlapping interval between the two particles. We calculate $\Delta v_{ij,\rm max}$ as a simple overestimate rather than the true

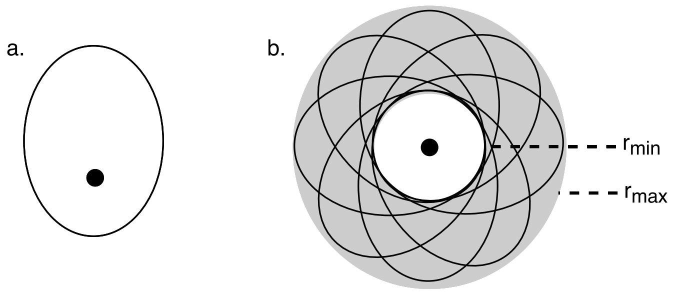

Figure 3. Panel a. shows a regular orbit in 2D, while panel b. shows the construction of a ring by rotating the orbit's pericenter by 2π.

maximum in order to speed up the calculation. As a result of ignoring the inclination and marginalizing over the phase and longitude of periastron, $\Delta v_{\rm max}$ can be calculated from just the semi-major axis and the eccentricity. We note that we do not need to solve Kepler's equation in estimating $\Delta v_{\rm max}$.

We can then calculate $f_{\rm H}$ by taking the maximum over all interacting particle pairs (see Fig 1),

$$f_{\rm H} = 4 \max_{i,j} \frac{\Delta v_{ij,\rm max} dt}{r_{H,i} + r_{H,j}}.$$

The numerical constant 4 ensures that two particles move at most one quarter of $f_{\rm H}(r_{H,i}+r_{H,j})$ in one timestep and therefore no close encounters are missed.

In practice, we also round up $f_{\rm H}$ to the nearest 1.25^x where x is an integer to avoid continuous fluctuations in $f_{\rm H}$. By default, our adaptive $f_{\rm H}$ algorithm is enabled in HERMES. Since the algorithm assumes particles move on approximately coplanar orbits, it may therefore fail at very high mutual inclinations. We note that the above algorithm chooses the smallest value of $f_{\rm H}$ that captures all close encounters; however, small values of $f_{\rm H}$ introduce numerical errors when switching between integrators (see Section 3). Therefore, to avoid large errors we set a default lower limit of $f_{\rm H} = 3$, which is close to the default close encounter boundary for MERCURY and SyMBA. The user can specify their own lower limit for $f_{\rm H}$ by setting `ri_hermes.hill_switch_factor` at runtime.

The adaptive $f_{\rm H}$ algorithm can be switched off by setting the variable `ri_hermes.adaptive_hill_switch_factor` to zero. If the adaptive $f_{\rm H}$ algorithm is switched off, setting `ri_hermes.hill_switch_factor` simply defines a constant value of $f_{\rm H}$ for the duration of the simulation (analogous to MERCURY and SyMBA).

We decided against devising a similar algorithm for $f_\odot$ due to the additional difficulties that can arise. For example, an object in a circular orbit around a planet would be confused as a heliocentric orbit with a very high eccentricity, leading to large relative velocities and an excessive $f_\odot$ value.

3 ERROR

Several terms contribute to the relative energy error of an integrator (e.g. Rein & Spiegel 2015):

$$E = E_{\rm floor} + E_{\rm round} + E_{\rm bias} + E_{\rm scheme}. \quad (1)$$

$E_{\rm floor}$ is a constant due to the inability to represent numbers with arbitrary precision on a computer. Here we work exclusively in IEEE754 double floating point precision and thus have $E_{\rm floor} \sim 10^{-16}$.

$E_{\rm round}$ arises when a computation is performed on two floating point numbers. Almost all operations (addition, multiplication, square roots) lead to a roundoff error at the level of the machine precision. The IEEE754 standard guarantees that the round-off error in consecutive floating point operations is random, thus leading to a $\propto t^{1/2}$ growth of $E_{\rm round}$ with time.

$E_{\rm bias}$ is the error from any biased operations and grows at least as $\propto t$. Biased operations can originate from poor implementations[3] or from library functions that the IEEE754 standard does not guarantee will return unbiased results[4] .

$E_{\rm scheme}$, the final term in Eq. 1, is the error introduced by the algorithm itself. Typically, this quantity is bound for symplectic integrators but grows linearly with time for non-symplectic integrators.

The important question is which error term dominates, and the answer will depend on the problem at hand. For example, if a three-body system (star and two planets) is integrated with IAS15, $E^{ias}_{\rm scheme} \approx 10^{-28}$ and the dominant error term will be $E_{\rm round}$, starting at 10^{-16} and growing as $t^{1/2}$ (see Rein & Spiegel 2015). If we instead integrate the system with WHFast, the dominant error term will be $E_{\rm scheme}$, which is determined both by the mass ratio in the system and the timestep. For typical parameters $E^{WH}_{\rm scheme} \sim 10^{-9}$, and only for very long simulation times ($\sim 10^{14}$ timesteps) will the growth of $E_{\rm round}$ dominate over $E^{WH}_{\rm scheme}$. For biased implementations, $E_{\rm bias}$ will dominate the error budget at earlier times.

For typical simulations integrated with HERMES, $E_{\rm scheme}$ will dominate. The WH algorithm integrates a slightly different Hamiltonian from the true Hamiltonian described by the system, leading to an error that is constant as long as the integrated Hamiltonian remains constant as well. However each time a particle is transferred to or from the global simulation (see Section 2.3), the WH-integrated Hamiltonian changes, and thus the error will change too. For a typical WH integration, working in democratic heliocentric coordinates, $E_{\rm scheme}$ is (e.g. Saha & Tremaine 1994; Wisdom 2006):

$$E^{\rm WH}_{\rm scheme} = \frac{dt^2}{12}\left\{\left\{H_K, H_\beta\right\}, 0.5H_K + H_\beta\right\} + O(dt^4) \quad (2)$$

Here H_K is the Keplerian Hamiltonian, $H_\beta = H_C + H_I$ is the summed momentum cross-term and interaction Hamiltonians, respectively, and {} are Poisson brackets (the quantities are explicitly defined for a test case with three particles below). For large N-body systems, Eq. 2 quickly becomes very difficult to evaluate analytically. However, we can gain some insight by applying Eq. 2 to a simple system.

We turn to a three body problem consisting of a star (active body), planet (active body) and planetesimal (semi-active body), shown in Figure 4. The planetesimal is initially placed inside $f_{\rm H}$ with sufficient velocity such that the distance between the planet and planetesimal grows over time. While the planetesimal is inside $f_{\rm H}$ (panel a. in Fig. 4) the system is integrated to machine precision by IAS15. However once the planetesimal leaves $f_{\rm H}$ (panel b. in Fig. 4) the system switches to being integrated by WHFast, and an error of size $E^{\rm WH}_{\rm scheme}$ is introduced.

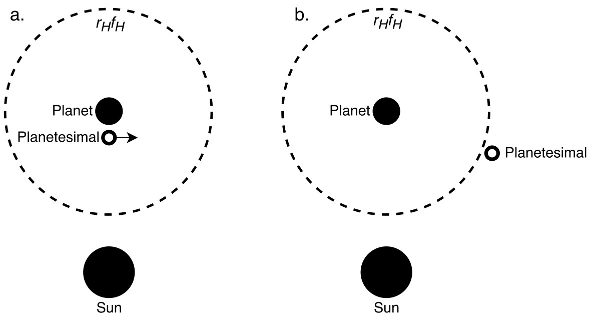

Figure 4. Three body problem, in the reference frame of the planet. In a. the initial setup is shown, where the planetesimal starts near the planet, inside a sphere of radius $r_H f_{\rm H}$ and the entire system is integrated purely by IAS15. Here the arrow indicates the initial direction of the planetesimal. In b. the planetesimal exits the sphere with radius $r_H f_{\rm H}$ and the system is integrated purely via WHFast, introducing a numerical error of $E^{\rm WH}_{\rm scheme}$.

To estimate $E^{\rm WH}_{\rm scheme}$, we start from the general Hamiltonian for an N-body system in Democratic Heliocentric coordinates:

$$H = H_0 + H_K + H_C + H_I$$

where H_0 is a constant describing the motion of the centre of mass along a straight line, and disappears when we evaluate Eq 2. The remaining terms in Eq. 3 take the form (Duncan et al. 1998):

$$H_K = \sum_{i=1}^{N-1} \frac{\mathbf{P_i^2}}{2m_i} - \sum_{i=1}^{N-1} \frac{Gm_0 m_i}{|\mathbf{Q_i}|} \quad (3)$$

$$H_C = \frac{1}{2m_0}\left|\sum_{i=1}^{N} \mathbf{P_i}\right|^2 \quad (4)$$

$$H_I = -\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \frac{Gm_i m_j}{|\mathbf{Q_i} - \mathbf{Q_j}|} \quad (5)$$

where the canonical coordinates $\mathbf{Q}$ and $\mathbf{P}$ are:

$$\mathbf{Q_i} = \begin{cases} \mathbf{r_i} - \mathbf{r_0} & \text{if } i \neq 0 \\ \frac{1}{m_{\rm tot}} \sum_{j=0}^{N} m_j r_j & \text{if } i = 0 \end{cases} \quad (6)$$

$$\mathbf{P_i} = \begin{cases} \mathbf{p_i} - \frac{\mathbf{m_i}}{\mathbf{m_{tot}}} \sum_{\mathbf{j=0}}^{\mathbf{N}} \mathbf{p_j} & \text{if } i \neq 0 \\ \sum_{j=0}^{N} \mathbf{p_j} & \text{if } i = 0 \end{cases} \quad (7)$$

Here $\mathbf{p}$, $\mathbf{r}$, m are a particle's momentum, position and mass in any inertial frame respectively, while G is the gravitational constant and $m_{\rm tot} = \sum_{j=0}^{N} m_j$ is the total mass of the system. $\mathbf{Q_i}$ are therefore heliocentric positions (with $\mathbf{Q_0}$ the centre of mass), while $\mathbf{P_i}$ are barycentric momenta (with $\mathbf{P_0}$ the momentum of the centre of mass).

[3] For example, the expression `x*(2./3.)` multiplies x by a number that is consistently slightly too big or too small when represented in binary. By contrast, the expression `2.*(x/3.)` multiplies and divides x by numbers that are exactly representable in binary and is unbiased.
[4] For example, the standard library function `sqrt()` returns an unbiased result whereas `sin()` returns a biased result.

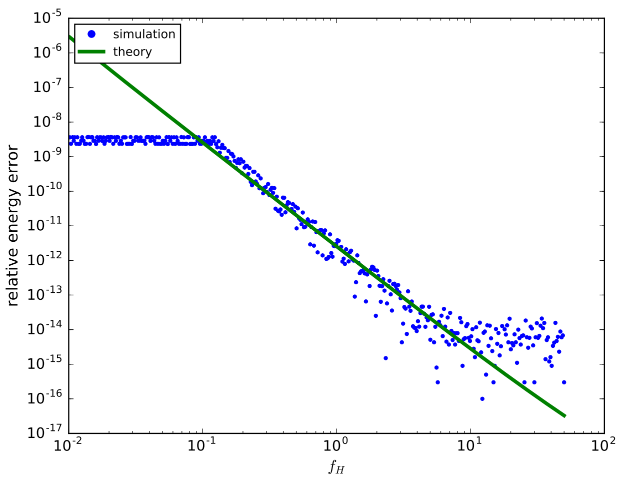

Figure 5. Final relative energy error as a function of f_{H} for a star-planet-planetesimal system. Blue dots are numerical simulations, the green curve is the theoretical prediction of Eq. 8.

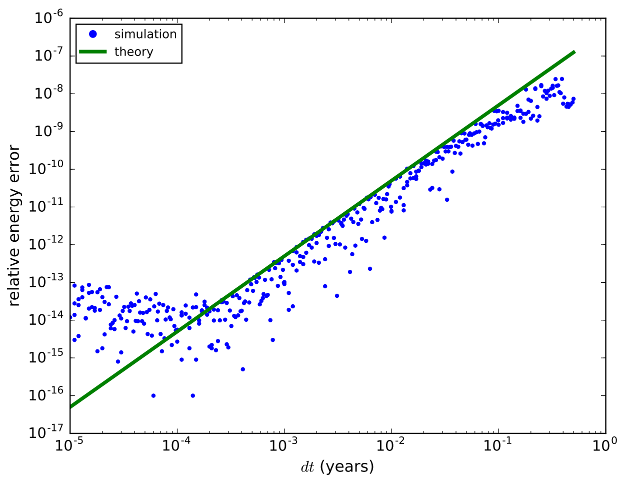

Figure 6. Final energy error as a function of dt for a star-planet-planetesimal system. Blue dots are numerical simulations, the green curve is the theoretical prediction of Eq. 8.

We also further simplify the system to two dimensions by considering motion in a plane. One can then straightforwardly, albeit tediously, evaluate the Poisson bracket in Eq. 2 by plugging in Eqs. 3–5 and taking derivatives with respect to all three particles. We make the further simplifying assumptions that $m_0 \gg m_2 \gg m_1$ and that $v_1 \approx v_2 \approx \sqrt{Gm_0/a}$, where v denotes particle velocities and a_2 is the semi-major axis of the planet. In addition, after solving Eq. 2 we set the distance of the planetesimal from the planet to $r_H f_{\mathrm{H}}$, where r_H is the Hill radius of the planet. This is true at the moment the integration method is switched. We are then left with a single dominating term,

$$E^{\mathrm{HERMES}}_{\mathrm{scheme}} \approx \frac{dt^2}{12} \frac{G^2 m_0 m_1 m_2}{a_2 (r_H f_{\mathrm{H}})^3}, \tag{8}$$

where we have ignored numerical constants of order unity. An IPython notebook with a computer derivation is available at `https://github.com/silburt/hermes_ipython`.

We compare our theoretical predictions in Eq. 8 to numerical tests in Figures 5 and 6. Our numerical setup consists of a star with mass $1M_\odot$, a Neptune mass planet on a circular orbit at 1 AU, and a planetesimal with mass $10^{-8}M_\odot$ placed at 0.001 AU from the planet. To marginalize over the phase of the encounter when sampling the energy error, the initial position of the planetesimal is randomized for each realization. In addition, the planetesimal is given a small kick equal to the escape velocity of the planet to ensure that the planetesimal-planet distance increases with time. Each realization is simulated for 7 years. When varying the timestep in Fig. 6 we use a constant $f_{\mathrm{H}} = 6$, and when varying f_{H} in Fig. 5 we use a constant timestep of $dt = 0.058$ days.

In Figure 5 one can see that the numerical tests agree with the theoretical predictions of Eq. 8. In addition, one can see the two extreme regimes on each end of the figure. For particles starting outside the $r_H f_{\mathrm{H}}$ boundary the integrator always uses `WHFast` (like panel b. of Fig. 4), while for particles inside the $r_H f_{\mathrm{H}}$ boundary the simulation uses `IAS15` (like panel a. of Fig. 4). In Figure 6 one can see that the relative energy error is proportional to dt^2, again matching the predictions of Eq. 8. For both Fig. 5 and Fig. 6, we have not performed any kind of fit, we simply over-plotted Eq. 8 with the data from our numerical experiments. We have performed other suites of simulations testing how the energy error scales with all other relevant quantities (semi-major axis, planetesimal mass, planet mass, stellar mass) and find Eq. 8 in good agreement. An IPython notebook for these experiments (including tests of the other relevant quantities) is available at `https://github.com/silburt/hermes_ipython`.

We now extend the characterization of the error to a more realistic case with N particles. We refer to the total relative energy error for an integration as $E^{\mathrm{HERMES}}_{\mathrm{scheme,tot}}$. Since an error of size $E^{\mathrm{HERMES}}_{\mathrm{scheme}}$ is introduced each time a particle leaves/enters the global simulation, the total errors should be related to the number of close encounters, N_{CE}. In the ideal case where the integrator is unbiased, the error $E^{\mathrm{HERMES}}_{\mathrm{scheme}}$ introduced by each close encounter is random, and $E^{\mathrm{HERMES}}_{\mathrm{scheme,tot}}$ will grow as a $N_{\mathrm{CE}}^{1/2}$ random walk. However, if `HERMES` is biased (i.e. $E^{\mathrm{HERMES}}_{\mathrm{scheme}}$ is not random), then $E^{\mathrm{HERMES}}_{\mathrm{scheme,tot}}$ will grow faster than $N_{\mathrm{CE}}^{1/2}$.

Assuming the unbiased case, $E^{\mathrm{HERMES}}_{\mathrm{scheme,tot}}$ is equal to:

$$E^{\mathrm{HERMES}}_{\mathrm{scheme,tot}} = K \cdot E^{\mathrm{HERMES}}_{\mathrm{scheme}} \cdot \sqrt{N_{\mathrm{CE}}} \tag{9}$$

where K is a constant of proportionality. We test Eq. 9 against numerical tests for a Solar mass star, a Neptune mass planet on a circular orbit at 1 AU, and a disk of 200 planetesimals located between $0.98 - 1.02$ AU. The initial inclinations and eccentricities of the planetesimals in the disk are set to 0, while the argument of perihelion and true anomaly are drawn from a uniform distribution. In addition, for these simulations we set $f_{\mathrm{H}} = 6$ and $dt = 0.015$ years. We performed numerous integrations, integrating each realization for a randomly chosen number of orbital periods between 10-1000, yielding different numbers of close encounters.

The results are shown in Figure 7, the x-axis showing the

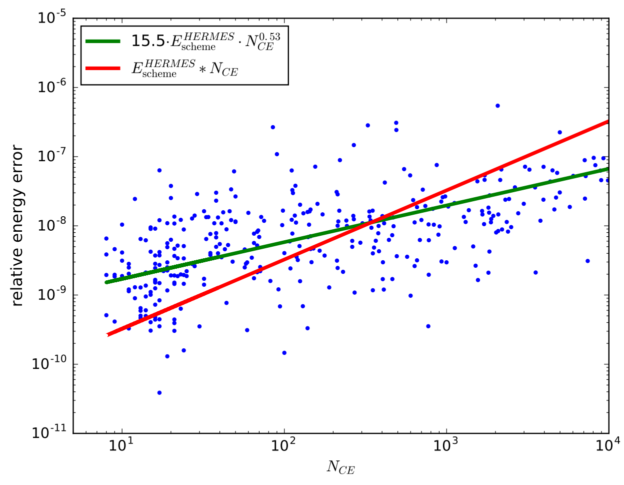

Figure 7. Final relative energy error as a function of the number of close encounters, for a system composed of a star, planet and 200 planetesimals. Blue dots are numerical simulations, the green line is our unbiased theoretical prediction of Eq. 9 with $K = 15$, while the red line is the biased theoretical prediction.

number of close encounters during a simulation and the y-axis showing the final relative energy error for each simulation. We fit a power-law distribution to the data using Python's Scipy Optimize Curve Fit package (Peterson 2009), displayed as a green line in Fig. 7. The resulting fit is $E^{\rm HERMES}_{\rm scheme,tot} = 15.5 \cdot E^{\rm HERMES}_{\rm scheme} \cdot N^{0.53}_{\rm CE}$, so the energy growth is well approximated by Eq. 9. For reference, we plot the biased prediction of $E^{\rm HERMES}_{\rm scheme,tot} \propto N_{\rm CE}$ as a red line. We conclude that HERMES is unbiased for this setup. Thus, Eq. 9 provides an intuitive way of understanding how the error of HERMES grows without having to analytically solve Eq. 2 in three dimensions for N particles.

The results from Fig. 7 did not allow for physical collisions between particles. When physical collision are enabled a systematic bias can be introduced.

To see why, note that for each close encounter two contributions to the energy error arise according to Eq. 8; one when the particles are transferred from the global to the mini simulation (i.e. the ingress of the close encounter) and one when the particles are transferred back from the mini to the global simulation (i.e. the egress of the close encounter). The precise energy change depends on the specific properties of the system (phases of the orbits, angles of approach, etc.), and typically the energy changes associated with the ingress and egress of the close encounter are anticorrelated. As a result, no appreciable energy error is introduced by close encounters, and the energy over the course of a simulation grows as expected according to Eq. 9.

However, when a physical collision occurs in HERMES, the close encounter only has an ingress, resulting in a biased growth in the energy error. In practice this energy bias is orders of magnitude smaller than the physical energy lost during a collision, and therefore should not interfere with the longterm evolution of a system. We plan to study this issue in more detail in the future.

4 EXAMPLES

We now highlight a number of possible simulations that can be performed using HERMES.

4.1 Massive Outer Solar System

Both Duncan et al. (1998) and Chambers (1999) simulated the outer Solar System, but increased the masses of all planets by a factor of 50 to trigger close encounters between planets. Analogous to Chambers (1999) and Duncan et al. (1998) we use $f_{\rm H} = 3$, $dt = 0.03$ yrs, and integrate the system for 1000 years.

We perform a number of simulations of the massive outer Solar System, and find that the relative energy error stays bounded at $\sim 10^{-7}$ for all simulations, matching the results of Chambers (1999) and Duncan et al. (1998).

4.2 Migration of a Planet in Planetesimal Disk (Kirsh et al. 2009)

Here we reproduce the results of Kirsh et al. (2009) for the migration of a single planet embedded in a planetesimal disk. In this study a 2.3$M_\oplus$ planet orbits a Solar mass star at 25 AU, embedded in a disk of $\sim 6 \cdot 10^4$ planetesimals. The planetesimal disk extends 10.5 AU on each side of the planet, each planetesimal has a mass 1/600th of the planet, and an overall surface density profile proportional to a^{-1} is used. The radii of all orbiting particles were determined assuming a constant density of 2 g/cm^3. The eccentricities and inclinations were drawn from a Rayleigh distribution, which is parameterized by the scale parameter σ. For this experiment we use $\sigma_e = 0.01$, and $\sigma_i = 0.005$, where inclination is in radians. The argument of periapse, true anomaly, and longitude of ascending node were all randomly drawn from uniform distributions over $[0, 2\pi]$. Adopting the default settings of Kirsh et al. (2009), we set HERMES to merge particles inelastically, set $dt = 2$ years, $f_{\rm H} = 5$, $f_\odot = 15$, and track the energy lost due to collisions or ejections. We also set `r->ri_hermes.adaptive_hill_switch_factor = 1`, activating the adaptive Hill switch routine (Section 2.5). An IPython notebook containing the code to run this example can be found at `https://github.com/silburt/hermes_ipython`.

We run 6 separate simulations, each for 70,000 years, and plot the results in Fig. 8. The relevant comparison plot in Kirsh et al. (2009) is the lower right panel in Figure 3. For all runs in both Kirsh et al. (2009) and our work the final position of the planet is between $18.5 < a < 20$ AU, and show similar evolution tracks throughout the simulation. The relative energy error over the course of all our simulations do not exceed $2 \cdot 10^{-7}$ after accounting for the energy lost in inelastic collisions.

4.3 Comparison to MERCURY and SyMBA– Long Simulations

Here we perform a set of long simulations and compare our results to MERCURY and SyMBA. These integrators are also capable of integrating complex N-body systems with close encounters, and also make use of semi-active particles. The sim-

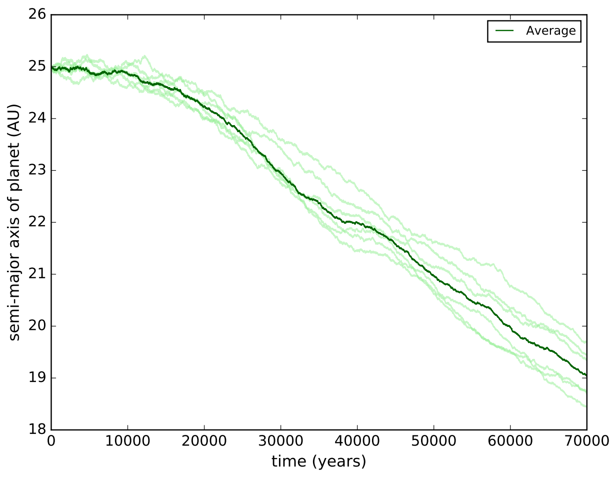

Figure 8. Planet's semimajor axis vs. time, analogous to the numerical experiment in the lower right panel of Figure 3 from Kirsh et al. (2009). Light green lines represent individual runs, while the dark thicker green line represents the average of the individual runs.

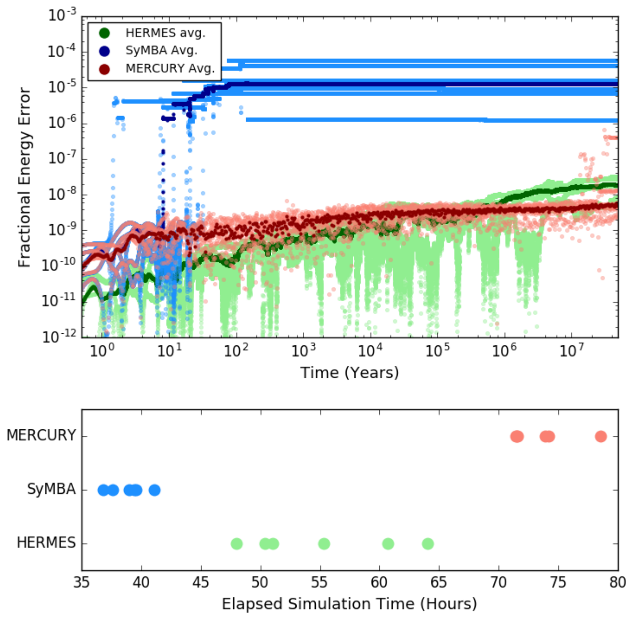

Figure 9. A test of HERMES, MERCURY and SyMBA for collections of 50Myr, 50 planetesimal runs. Top panel shows the relative energy error over time, with individual runs in lighter shades and averaged values in dark shades. Bottom panel shows the elapsed simulation times of individual runs, using the same colour scheme as the top panel.

ulations for these tests contain a Solar-mass star, a Neptune-mass planet at $a = 1$ AU and a disk of 100 semi-active planetesimals distributed according to a powerlaw between $0.8 - 1.2$ AU. The mass of each planetesimal is a third of a lunar mass, the eccentricities and inclinations are set to 0, and the argument of periapse, true anomaly, and longitude of ascending node were all randomly drawn from uniform distributions over $[0, 2\pi]$. We set $dt = 0.01$, $f_{\rm H} = 3$, merge particles inelastically, and use our adaptive $f_{\rm H}$ routine (Section 2.5). In addition, for all three integrators we track the energy lost due to inelastic collisions and ejections so that we can isolate the numerical energy error. We accomplish this by using the `eoffset` variable in `SyMBA` and `EN(3)` variable in `MERCURY`, and calculate the relative energy error according to $(E_i + E_{\rm off} - E_0)/E_0$, where E_i is the total energy at iteration i, E_0 is the initial total energy and $E_{\rm off}$ is the energy lost due to collisions and ejections, i.e. `eoffset` in `SyMBA` and `EN(3)` in `MERCURY`.

We feed identical initial conditions to `SyMBA`, `MERCURY` and `HERMES`, and evolve all simulations for 50 Myr. We run 6 simulations per integrator that differ only by the seed of the random number generator. We average the relative energy error for these simulations to smooth out variations between individual runs. The results are presented in Figure 9. The top panel displays the relative energy error with each integrator, while the bottom panel shows the elapsed time for each individual run.

Looking at the top panel, `SyMBA` incurs significant energy jumps early in the simulation. We suspect these energy jumps are due to inadequately resolved close encounters, since the energy jumps are uncorrelated with particle collisions. We simulated runs using different combinations of `RHSCALE` and `RSHELL`, i.e. the parameters which define close encounter regions, but were unable to resolve these energy jumps. By the end of the simulation, `SyMBA` is orders of magnitude less accurate than `MERCURY` and `HERMES`.

We see that `HERMES` initially has the lowest energy error, but grows faster than `MERCURY`. The final energy error for an integration therefore depends on the particular problem (length of simulation, timestep, number and severity of close encounters etc.). Furthermore, we find that some `MERCURY` simulations undergo significant energy jumps, probably also due to very close encounters that are not properly resolved. Our adaptive $f_{\rm H}$ algorithm (Section 2.5) protects against these situations from occurring, and for a properly chosen $f_{\rm H}$ and dt combination we find the energy growth of `HERMES` to be well behaved.

The bottom panel shows that for these simulations `HERMES` is slower than `SyMBA` but faster than `MERCURY`. We see that `HERMES` exhibits a larger variance in elapsed simulation times than `MERCURY` and `SyMBA`. This is because when the adaptive $f_{\rm H}$ routine is engaged, $f_{\rm H}$ can be enlarged considerably during severe encounters, slowing down the integrator. In summary, these simulations show that `HERMES` provides a good balance between speed and accuracy.

5 CONCLUSION

In this paper we have presented `HERMES`, a hybrid integrator capable of integrating close encounters and collisions. `HERMES` integrator is composed of two parts, a global simulation which contains all particles and a mini simulation which contains all active particles (e.g. stars and planets) plus any semi-active/test particles (e.g., planetesimals) involved in a close encounter. The global simulation is integrated with WHFast while the mini simulation is integrated with IAS15. Comparing HERMES to the openly available MERCURY and SyMBA we find that HERMES provides a good balance between accuracy and speed.

HERMES takes a more direct approach when integrating close encounters over other methods. However this has enabled us to characterize the error of HERMES, and we find that the switching error from a single close encounter is well described by Eq. 8, which we calculate from first principles. The total energy error of HERMES for an N-body simulation is well described by a random walk, (see Eq. 9) with the step size being the switching error.

We have also developed an adaptive algorithm that chooses the optimal Hill switch factor, $f_{\rm H}$, which governs the size of the close encounter region surrounding each particle. This frees the user from optimizing integrator parameters for typical cases. Finally, we have introduced a number of new features in REBOUND, including a new heliocentric version of WHFast, semi-active particles (see Fig. 1) and inelastic collisions.

We have showcased a number of problems well-suited for HERMES, and compared our integrator's performance to similar integrators in the literature. In particular, we integrated the outer Solar System with planetary masses increased by a factor of 50, and we simulated a planet migrating through a disk of planetesimals, both in the limit of many planetesimals ($\sim 10^5$) and short times ($\sim 10^3$ orbits), and few planetesimals (100) and long times ($\sim 10^8$ orbits). Many more types of problems are possible with HERMES, and additional examples can be found at `https://github.com/hannorein/rebound`.

REFERENCES

Chambers, J. 1999, MNRAS, 304, 793

Duncan, M. J., Levison, H. F., & Lee, M. H. 1998, AA, 116, 2067

Kirsh, D. R., Duncan, M. J., Brasser, R., & Levison, H. F. 2009, AA, 199, 197

Peterson, P. 2009, Int. J. of Computational Science and Engineering, 4, 296

Press, W., Flannery, B., Teukolsky, S., & Vetterling, W. 1988, Numerical Recipes in C (Cambridge University Press)

Rein, H., & Liu, S.-F. 2012, AA, 537, 10

Rein, H., & Spiegel, D. S. 2015, MNRAS, 446, 1424

Rein, H., & Tamayo, D. 2015, MNRAS, 452, 376

Saha, P., & Tremaine, S. 1992, AJ, 104, 1633

—. 1994, AA, 108, 1962

Wisdom, J. 2006, AJ, 131, 2294

Wisdom, J., & Holman, M. 1991, AJ, 102, 1528

Yoshida, H. 1990, Science Direct, 150, 6

—. 1993, AA, 56, 16