ORIGINAL ARTICLE

The detection of relativistic corrections in cosmological N-body simulations

Jean-Pierre Eckmann[1] ⬤ · **Farbod Hassani[2]** ⬤

Abstract
Cosmological N-body simulations are done on massively parallel computers. This necessitates the use of simple time integrators and, additionally, of mesh-grid approximations of the potentials. Recently, Adamek et al. (Nat Phys 12:346, 2016), Barrera-Hinojosa and Li (GRAMSES: a new route to general relativistic *N*-body simulations in cosmology—I. Methodology and code description, 2019) have developed general relativistic N-body simulations to capture relativistic effects mainly for cosmological purposes. We therefore ask whether, with the available technology, relativistic effects like perihelion advance can be detected numerically to a relevant precision. We first study the spurious perihelion shift in the Kepler problem, as a function of the integration method used and then as a function of an additional interpolation of forces on a two-dimensional lattice. This is done for several choices of eccentricities and semi-major axes. Using these results, we can predict which precisions and lattice constants allow for a detection of the relativistic perihelion advance in N-body simulation. We find that there are only small windows of parameters—such as eccentricity, distance from the central object and the Schwarzschild radius—for which the corrections can be detected in the numerics.

1 Introduction

We consider here so-called (cosmological) N-body simulations such as Adamek et al. (2016a), Springel (2005), Teyssier (2002). In these numerical studies, potentials between the (many) particles are computed on a lattice (mesh-grid) because of the way such calculations are implemented on supercomputers. Additionally, some of these projects (such as Adamek et al.

✉ Jean-Pierre Eckmann
   jean-pierre.eckmann@unige.ch

   Farbod Hassani
   farbod.hassani@unige.ch

[1] Département de Physique Théorique and Section de Mathématiques, University of Geneva, Geneva, Switzerland

[2] Département de Physique Théorique, University of Geneva, Geneva, Switzerland

2016b; Barrera-Hinojosa and Li 2019) add relativistic corrections to the forces and therefore to the trajectories of particles. The aim of our study is to give bounds on the detectability of these effects, given the computational restrictions of these large-scale projects. We will see that in many current simulations, the necessary precision to detect relativistic effects on the orbits of particles can simply not be achieved.

It is of course not difficult to devise codes which will compute the perihelion advance under relativistic corrections to arbitrary high precision. It is not the aim of our paper to study such algorithms, but rather, to see how well the integration algorithms work in the N-body simulations. In these simulations, because one considers essentially a gas of many particles, the user is restricted to rather standard integration methods, which just use the differential equations, but necessarily cannot make use of the many invariants known for the (non-relativistic) Kepler problem, see, e.g., Preto and Saha (2009). Therefore, we need to first study the performance of standard integration schemes, such as Euler, Runge–Kutta, and leapfrog, because these are the methods which are widely used. We will see that only with very high precision one is able to detect the (usually quite small) relativistic corrections. Once this has been done, we can turn our attention to the effects of the discretizations (of space), which give then bounds on the necessary grid constants for which relativistic effects could be detected. We will determine parameter regions where the relativistic effects can be detected and show that, most often, these regions are quite small.

By starting with the simple Kepler problem, in $\mathbb{R}^2$, we can concentrate on the different numerical effects in a systematic and clean way. Even so, the reader should realize that there are several quantities to be considered. The first is the numerical precision of the time integrator. We study it here in the context of the subroutines in ODEX (Hairer et al. 1993), and we also compare it to other methods, such as Euler, leapfrog (Verlet–Störmer) (Hairer et al. 2003) or Runge–Kutta with fixed time step.

To do this, we quantify the numerical errors on trajectories of particles revolving around a central object. This will allow us to give conditions which ascertain which orbits in a specific N-body simulation are precise enough to be able to measure the general relativistic perihelion[1] shift.

After this, we consider the particle-mesh N-body scheme, as is widely used, see, e.g., Adamek et al. (2016a), Springel (2005). In it, forces (coming from fields and potentials) are discretized and represented on a lattice. Such elements Arnold (2002) are then used to compute the values of the fields at the particles' positions.

The force interpolation approximations are usually piecewise differentiable and, depending on the implementations mentioned above, use different elements. It is clear that if the mesh size of the approximation (of the force) goes to 0, so will the error. But the relevant question here is to quantify what kind of phenomena can be captured, given the numerous hardware and modeling constraints.

Particle-mesh N-body simulations are used to study the evolution of particles under gravity. These codes can be used to study systems at different scales, from cosmological scales to the size of the solar system, as the methods and forces are appropriate for all scales. In the particle-mesh N-body scheme (Springel 2005; Adamek et al. 2016a), space-discretizations are performed to take care of the large number of particles. We analyze two common force interpolations which are used for N-body simulations purposes, namely the so-called linear and bilinear methods, which respectively correspond to first- and second-order interpolation. We will see that, under conditions to be specified, the effect of spatial discretization can be quite large and sometimes depends on the angle θ between the direction of the perihelion and

---

[1] We use "perihelion" even if the central mass is not the Sun.

the axes of the discrete lattice. This happens when the discretization produces discontinuous forces, i.e., for the first-order force interpolation. In this case, the maximal errors are proportional to the lattice constant dx. On the other hand, in the second-order force interpolation, when one varies θ, there is a small perihelion shift, fluctuating around 0. These fluctuations are seen to be of order $dx^{1.3}$. Due to the highly nonlinear step size of ODEX, we are not able to derive analytically this size of the fluctuations.

Our numerical tests show that, unless the discretization is extremely fine, the system will show an uncertainty of the perihelion, for the Kepler problem, for both force interpolation methods. Our calculations give limits on the detectability of relativistic effects, as a function of method, lattice spacing, as well as eccentricity and the relativistic parameter $\Upsilon \equiv r_{\text{sch}}/r_{\text{per}}$, the ratio of the Schwarzschild radius of the central mass r_{sch} to perihelion distance of the orbit r_{per}.

2 Using standard time integrators

Our main interest is the detectability of general relativistic effects in N-body simulations, and in particular the study of discretization effects. But we first need to be sure that the time integration which is used in these projects does not already destroy the precision of the result more than the effect of the space discretization. This is the subject of this section.

A Hamiltonian problem can be integrated either as a motion in Euclidean space, or one can exploit the underlying symplectic structure of the problem. Of course, there are very good symplectic integrators, Hairer et al. (2006), but we decided not to use them, for two reasons.

The first is that in the particle-mesh N-body codes, the Euclidean approach is used to solve the system including particles and the fields on the lattice coordinates and expressing the system in symplectic coordinates is difficult. Second, as noted in Hairer et al. (2003) even the symplectic methods do not preserve the Runge–Lenz–Pauli vector (the orientation of the semi-major axis). This means that because of our focus on the relativistic perihelion advance, the symplectic integrators present no particular advantage. So we will stick with the classical high-order Runge–Kutta integrator, ODEX (Hairer et al. 1993). Because it allows for "continuous output," we can use it to determine easily the advance of the perihelion with high precision.

We summarize those properties of ODEX which are relevant for our study. As we will be working with elements to interpolate forces, we need to explain how the algorithm deals with discontinuities. This is illustrated in Hairer et al. (1993, Chapter II.9 and in particular, Fig. 9.6). In the interior of a plaquette, the algorithm chooses a high enough order to reach the required tolerance with a large time step h. On approaching the singularity, the algorithm lowers the order (to 4) but decreases h. In fact, the jump is approximated by a polynomial of degree 4, and this defines something like a new initial condition across the discontinuity. In the case of Runge–Kutta with fixed time step, the paper Back (2005) shows that there is a mean systematic error across the jump, which can be viewed as a weighted combination of evaluations of the vector field across the singularity.

3 Perihelion variation for time integrators with fixed time step

Here, we study the precision of perihelion calculations when working with *exact* forces. In later sections, we will then see how the grid discretization further affects this precision.

Certainly, if we want to discriminate between non-relativistic and relativistic effects, already the integration with exact forces needs to be precise enough. This will force us to choose a small enough time step h.

We analyze the perihelion shift for several standard time integrators with fixed time step, namely Euler, Newton–Störmer–Verlet-leapfrog, and second-order and fourth-order Runge–Kutta.

We call the time step h and we solve the Kepler problem in the form[2]

$$\ddot{x}(t) = F(t, x)/m, \tag{3.1}$$

with x, v, and F in $\mathbb{R}^2$ and m the mass of the object. For the convenience of the reader, we spell out these well-known methods.

Euler method   In this case, we solve (3.1) in the form

$$v_{n+1} = v_n + a_n h,$$
$$x_{n+1} = x_n + v_{n+1} h,$$

where v_n is the velocity vector at time step (n) which is defined as $v_n \equiv (x_n - x_{n-1})/h$ and a_n is the acceleration and is defined as the ratio of the force to mass $a_n \equiv a(t_n, x_n) = F(t_n, x_n)/m$. It is well known that this implicit/backward Euler method is more stable than the explicit/forward Euler method. But it is somewhat more difficult to implement for nonlinear differential equations.

Newton–Störmer–Verlet-leapfrog method   For this widely used method (sometimes called "kick-drift-kick" form of leapfrog) (Hairer et al. 2003) [Eq. 1.5], the updates are

$$v_{n+1/2} = v_n + a_n \frac{h}{2},$$
$$x_{n+1} = x_n + v_{n+\frac{1}{2}} h,$$
$$v_{n+1} = v_{n+\frac{1}{2}} + a_{n+1} \frac{h}{2}.$$

This method is used more often as it is a symplectic method and stable and is shown to work very well for various stiff ODEs (Hairer et al. 2003).

Second-order Runge–Kutta method   Finally, we will comment on the perihelion advance for the second- and fourth-order Runge–Kutta methods (with fixed time step h). The Kepler problem using second-order Runge–Kutta algorithm—also known as midpoint method—reads,

$$k_x^{(1)} = v_n,$$
$$k_v^{(1)} = a_n,$$
$$k_x^{(2)} = v_{n+\frac{1}{2}},$$
$$k_v^{(2)} = a_{n+\frac{1}{2}},$$
$$x_{n+1} = x_n + k_x^{(2)} h,$$
$$v_{n+1} = v_n + k_v^{(2)} h,$$

where $k_x^{(1)}$ is the estimate of velocity (derivative of x) in time step n, $k_v^{(1)}$ is the estimate of acceleration (derivative of v) in time step n and the same for $k_x^{(2)}$ and $k_v^{(2)}$. The acceleration at time $n + \frac{1}{2}$ is obtained by

---

[2] All positions, velocities, and the like are in $\mathbb{R}^2$.

$$a_{n+\frac{1}{2}} \equiv \frac{F(t_{n+\frac{1}{2}}, x_{n+\frac{1}{2}})}{m} = \frac{F(t_{n+\frac{1}{2}}, x_n + k_x^{(1)} h/2)}{m}.$$

Also, to obtain the velocity at time $n + \frac{1}{2}$ we need to use $k_v^{(1)}$. The corresponding tableau for the second-order Runge–Kutta method for each first-order differential equation is

$$\begin{array}{c|cc} 0 & & \\ 1/2 & 1/2 & \\ \hline & 0 & 1 \end{array}.$$

Fourth-order Runge–Kutta method The Kepler problem using forth-order Runge–Kutta method is basically the same as second-order Runge–Kutta, but with three points instead of one point in between to solve the position and velocity. The tableau we use for this method is Butcher (1963)

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}.$$

3.1 Results for various integration schemes

We solve the Kepler problem with the methods described above and find the perihelion variation for different time steps h. To determine the perihelion point of the orbit, we choose several points near the minimum distance to the central object after each revolution. Then we fit a parabola (for Runge–Kutta, we take a fourth-order polynomial to find the point closest to the central mass) and the minimum of the parabola is taken as the perihelion point. Figure 1 illustrates how this is done, for the particular example of Mercury. The red point is the perihelion. The spurious shift of the perihelion of Mercury using second-order Runge–Kutta method with $h = 0.00625$, which is the case considered in Fig. 1, is $\sim 7.8 \times 10^{-5}$ radians. We measure the positions in units Giga meters (Gm $\equiv 10^9$ m), time in Mega seconds (Ms $= 10^6$ sec) and masses in $M_{\text{earth}} = 5.972 \times 10^{24}$ Kg. In these units, the initial position (at the perihelion) is $46.001272(\cos(\theta), \sin(\theta))$ where θ is the angle between x-axis and semi-major axis. The initial velocity is perpendicular to the line connecting Mercury and the Sun, with magnitude 58.98. The potential is $-GM/r$ with $GM = 132733$ measured in the code's units.[3] When we will study the problem on the lattice, the angle θ will be important. In Fig. 2, the magnitude of perihelion variation for the different time integrators and the step size h is shown. The horizontal line shows the value of relativistic perihelion advance, and the green/red regions, respectively, show where the time integrator precision is/is not good enough to observe relativistic perihelion advance. Because time integrators over- or underestimate the perihelion, we plot its absolute deviation (which for Newton's law should be zero). This absolute value sets a limit of how small one has to take a time step h to be able to detect general relativistic corrections to the orbits.

The relativistic parameter $\Upsilon = r_{\text{sch}}/r_{\text{per}}$ for the Mercury–Sun case with $r_{\text{per}} = 46 \times 10^6$ km and $r_{\text{sch}} = 2.95$ km is $\Upsilon \approx 6.4 \times 10^{-8}$. The eccentricity of Mercury is ≈ 0.205630. Both parameters are considered small as the distance of Mercury to the Sun is much larger than the Schwarzschild radius of the Sun. We therefore consider also a more extreme case, where

---

[3] In units $\text{Gm}^3 \cdot M_{\text{earth}}^{-1} \cdot \text{Ms}^{-2}$.
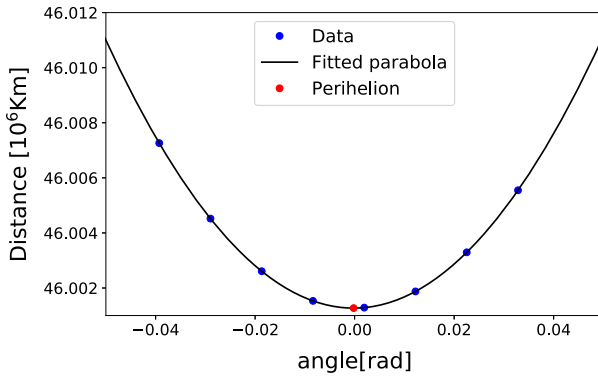
Fig. 1 Detecting the spurious perihelion change for the Kepler problem for the Mercury–Sun system. For each time step of integration, we determine the angle and the distance from the central body (the blue points), using the second-order Runge–Kutta method. We then fit a parabola through these points, and the minimum of the distance to the Sun is the red point (perihelion point). Note that the red point is very slightly to the left of 0 and shows the spurious perihelion shift due to the time integration imprecisions. We use this method to find the perihelion shift and to decrease the errors we take average over three revolutions

Fig. 2 Achievable precision for different integrators, as a function of step size h. Shown is the absolute value of the perihelion shift for the Mercury–Sun problem. To make relativistic corrections distinguishable, only points in the green region are good enough. The data points correspond to 1/3 of the advance after 3 rotations
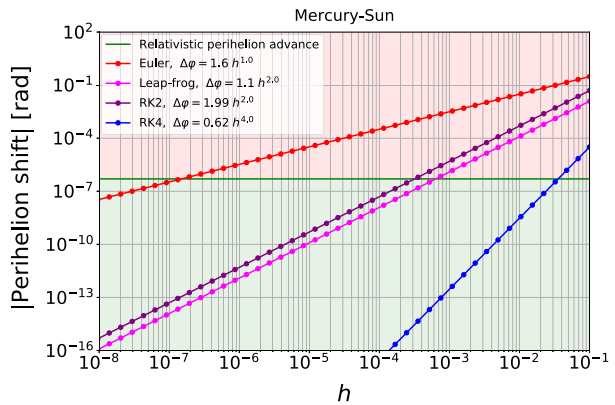

the relativistic effects are larger, such as the stars in the Galactic center known as the S-stars (Parsa et al. 2017) which are revolving around the central super massive black hole. For one of them, S2, the relativistic parameter at perihelion point is estimated from measurements to be $\Upsilon \approx 8.8 \times 10^{-5}$, and the eccentricity is 0.884. The details about the relativistic parameter and eccentricity can be found in "Appendix B."

In our numerical study, we cover therefore a large range of these parameters. Our results are summarized for the four integration methods in Fig. 2 as a function of the time step h. In Fig. 3, the comparison is done as a function of Υ/Υ_0, where Υ_0 is the relativistic parameter at perihelion point for the Mercury–Sun system. We also show the dependence on eccentricity.

4 Force interpolation

Having considered the numerics of the classical methods, we now study the effect of discretizing space. We again restrict attention to two dimensions and set, throughout, the lattice
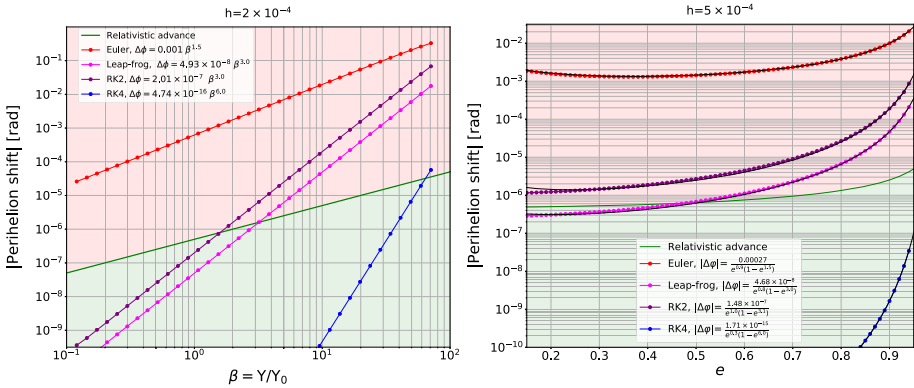
Fig. 3 Left—the absolute value of the perihelion shift as a function of the normalized relativistic parameter $\beta = \Upsilon/\Upsilon_0$, where Υ_0 is the relativistic parameter for Mercury–Sun. The red region shows where the method will fail to discriminate the relativistic perihelion advance from the integration errors (for the chosen step size of $h = 0.0002$). In the green region, one can safely use the method, for that specific orbit. When increasing β, the numerical perihelion shift increases, as according to (B.1) the velocity of the object in the perihelion point scales like $\sqrt{\Upsilon}$, while the perihelion distance scales like $\frac{1}{\Upsilon}$. In all the methods, the slope of the curve is higher than the slope of the relativistic advance curve $\Delta\varphi \sim \Upsilon$, which shows that for the orbits with large relativistic parameters, one has to choose the method and the time step very carefully. Right—the same representation as a function of eccentricity e. In all the methods, by increasing the eccentricity the numerical perihelion variation increases, as according to the (B.2) the velocity of the object and perihelion distance rescales, respectively, by $\sqrt{\frac{1+e}{1+e_0}}$ and $\frac{1-e}{1-e_0}$. In order to be able to measure the relativistic perihelion advance at each eccentricity, we need to use the method with the appropriate step size; for example, Euler and second-order Runge–Kutta do not work for any eccentricity, while leapfrog is good for $e \lesssim 0.5$ and fourth-order Runge–Kutta works perfectly for all eccentricities. All data points correspond to 1/3 of the advance after 3 rotations

spacing equal to dx.[4] In particular, we study the two force interpolations (linear and bilinear) which are mainly used in N-body simulations, see, e.g., Springel (2005) and Adamek et al. (2016a), and for which we will present numerical results. A very useful systematic derivation of finite elements for derivatives and differential complexes can be found in Arnold (2002). The setup is as follows: We are given a potential Φ, in our case the Newtonian potential $\Phi(x_1, x_2) = -GM/\sqrt{x_1^2 + x_2^2} = -GM/r$, from which we want to derive the forces on the particles. In the bilinear (quadratic) method, the mesh is given by integer coordinates (in $\mathbb{Z}^2$), and we assume that Φ is known in all points (i, j), with $i, j \in \mathbb{Z}$.[5] The force at lattice point i, j is then approximated by a vector with components

$$f_{i,j}^{(x)} = \frac{\Phi_{i+1,j} - \Phi_{i-1,j}}{2},$$

$$f_{i,j}^{(y)} = \frac{\Phi_{i,j+1} - \Phi_{i,j-1}}{2}.$$

---

[4] Due to the discretization, the angular momentum vector might not be conserved and we might have 3D motion; here, we assume that the force perpendicular to the plane of motion vanishes.

[5] Finite elements are of course obtained more easily on triangular lattices, but, because of requirements of large parallel computations, we study the lattice $\mathbb{Z}^2$.

Fig. 4 Notation for a plaquette on the unit mesh in $\mathbb{R}^2$

$c = (i, j+1)$ $d = (i+1, j+1)$

(u, v)
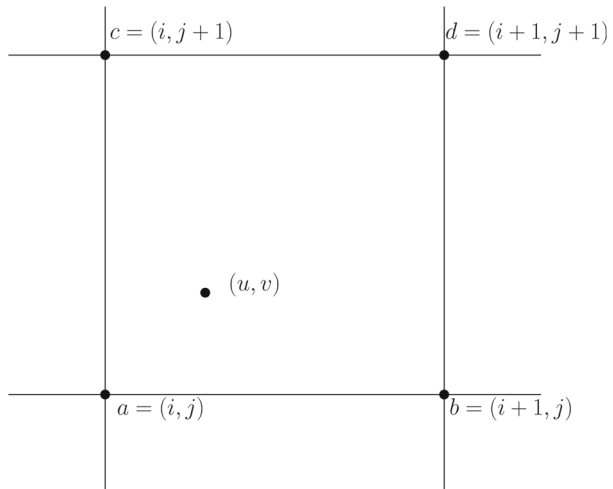
$a = (i, j)$ $b = (i+1, j)$

Note that the difference is taken over 2 mesh points around the point of interest. Assume that the point $x = (u, v) \in \mathbb{R}^2$ lies in the square with corners

$$a = (i, j), \quad b = (i+1, j), \quad c = (i, j+1), \quad d = (i+1, j+1),$$

cf. Fig. 4. We let $f_a^{(x)} = f_{i,j}^{(x)}$, and similarly for the other corners and the direction (y).

Let i be the integer part of u and let j be the integer part of v, and set $\xi = u - i, \eta = v - j$. The interpolated forces are then given by

$$F^{(x)}(u, v) = \left(f_a^{(x)}(1 - \xi) + f_b^{(x)}\xi \right) \cdot (1 - \eta) + \left(f_c^{(x)}(1 - \xi) + f_d^{(x)}\xi \right) \cdot \eta,$$

$$F^{(y)}(u, v) = \left(f_a^{(y)}(1 - \eta) + f_c^{(y)}\eta \right) \cdot (1 - \xi) + \left(f_b^{(y)}(1 - \eta) + f_d^{(y)}\eta \right) \cdot \xi. \quad (4.1)$$

Note that "c" and "b" change position between the x and y components. This interpolation method is called *bilinear method* as it is combination of two linear interpolations along the square, so it is a quadratic interpolation (Arnold 2002). This interpolation is continuous across the boundaries in both directions and for both components of the vector field. To verify this, one can for example restrict to the line connecting a and b. Then $\eta = 0$, and therefore one gets

$$F^{(x)}(u, v) = f_a^{(x)}(1 - \xi) + f_b^{(x)}\xi,$$

$$F^{(y)}(u, v) = f_a^{(y)}(1 - \xi) + f_b^{(y)}\xi.$$

The important thing is that the values only depend on a and b, but not on c and d and so continuity is guaranteed. The 3 other edges are similar.

In this scheme, as is well known, one needs 8 evaluations of Φ per plaquette. When the mesh size is dx instead of 1, all the calculations scale accordingly.

The other method, *linear method* which is also widely used in cosmological N-body simulations, e.g., Adamek et al. (2016a) is given, with similar notation—using g and G instead of f and F—by

$$g_{i,j}^{(x)} = \Phi_{i+1,j} - \Phi_{i,j},$$
$$g_{i,j}^{(y)} = \Phi_{i,j+1} - \Phi_{i,j},$$
$$G^{(x)}(u,v) = g_a^{(x)}(1-\eta) + g_c^{(x)}\eta,$$
$$G^{(y)}(u,v) = g_b^{(y)}(1-\xi) + g_d^{(y)}\xi,$$

with $a, \dots, d$ as before. This method is of lower order than the previous one and needs fewer evaluations. The advantage is that they need less memory, but of course, it is only 1st order.

Note that if (u,v) crosses the line connecting a and b, then $G^{(x)}$ is continuous, but $G^{(y)}$ has a jump discontinuity (of order about $\mathcal{O}(\Phi_{i,j} - \Phi_{i+1,j})$ when i and j are not too close to 0). Similar considerations hold on the other boundaries of the unit plaquette.

This scheme only needs 4 evaluations of Φ per two-dimensional plaquette, but the interpolation is not continuous. The Kepler problem can still be integrated numerically, but there will appear a spurious phase shift which is caused by the discontinuity. But the numerical errors again scale with the mesh size, albeit on a larger scale than in the first method.

We will now present the numerical results for these cases and then discuss the limitations they imply on trajectories in N-body simulations. Of course, often calculations are done in $\mathbb{R}^3$, resp. $\mathbb{Z}^3$, but for the study of numerical issues, two dimensions are enough. Restriction to one dimension is too easy, since the two methods coincide in that case.

5 Discretization versus relativistic perihelion advance

We have seen that high precision is needed to discriminate relativistic effects in the planar two-body problem. As several N-body codes use—in addition to the standard numerical integration schemes, a *discretization* of space—we now study the effects of these discretizations. To concentrate on them, we use a numerical integration of very high precision (ODEX, tolerance 8×10^{-11}) so that the effects described earlier are minimal, and the effect of discretization becomes visible.

As we want to measure the perihelion advance due to the discretization we stick to the general equations in which we do not use the symmetries of the Kepler problem. We just assume that the motion is on a plane, and we solve the equations for the relative distance between two masses assuming that $m \ll M$,

$$(\dot{x}, \dot{y}) = (v_x, v_y),$$
$$(\dot{v}_x, \dot{v}_y) = (H^{(x)}(u,v), H^{(y)}(u,v)),$$

where $H^{(x)}(u,v)$ is $F^{(x)}(u,v)$ or $G^{(x)}(u,v)$ as defined in Sect. 4.

We have, so far, analyzed in detail how much numerical precision is needed to detect general relativistic effects, as a function of eccentricity and the relativistic parameter Υ.

We start by presenting results for the bilinear interpolation (4.1). The simulations are done as follows: We take the parameters for Mercury, with initial position at $46.001 \cdot (\cos(\theta), \sin(\theta))$ and a velocity perpendicular to the Mercury–Sun line, of magnitude 58.98, in the counterclockwise direction. We require a tolerance of 8×10^{-11}, which is attainable with quadruple precision, using ODEX. For each value of θ, we determine the time for 1, 2, and 3 returns to the perihelion. The perihelion is found by looking for that angle where the distance from the Sun is minimal. This angle is found by linear and quadratic bisection, up to machine precision, using the "continuous output" from ODEX.[6] We repeat this for

---

[6] We use the standard algorithm "zeroin" of Dekker.
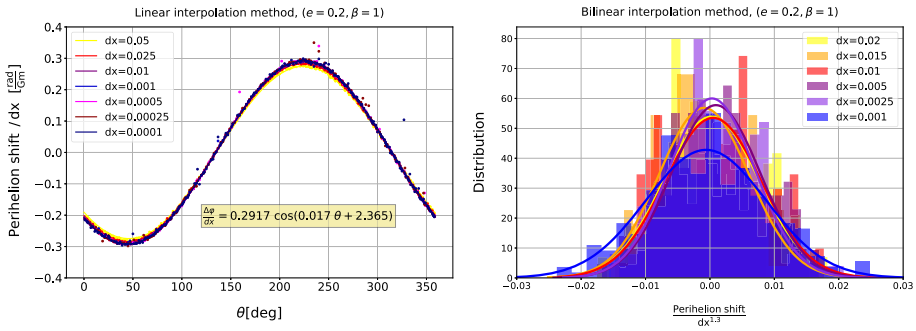
Fig. 5 Left—linear interpolation: the advance of the perihelion due to the discretization effect depends on the initial angle of the perihelion. For each value of dx, we show the deviation in radians, divided by dx. The curves clearly coincide. This shows that the deviations scale linearly with dx. Right—bilinear interpolation: We consider the perihelion shifts, divided by $dx^{1.3}$, for 180 equally spaced initial angles of the orbit. The bar graphs show the distribution of these quantities, for various choices of dx. We see that they obey a Gaussian fit (the solid lines). This shows that the shifts are random

180 initial angles covering 360 degrees in steps of 2 degrees, and this gives us 3×180 data points.[7] In Figs. 5 and 6, we show the results for several values of dx, for linear and bilinear approximation. Further inspection shows that these distributions are close to Gaussian, but the variance is somewhat smaller than dx, actually $dx^{1.3}$ is a reasonable approximation. As we mentioned before, an analytic estimate of this variance is difficult, because ODEX works with variable step size and order, with quite dramatic changes near the edges of the plaquettes.
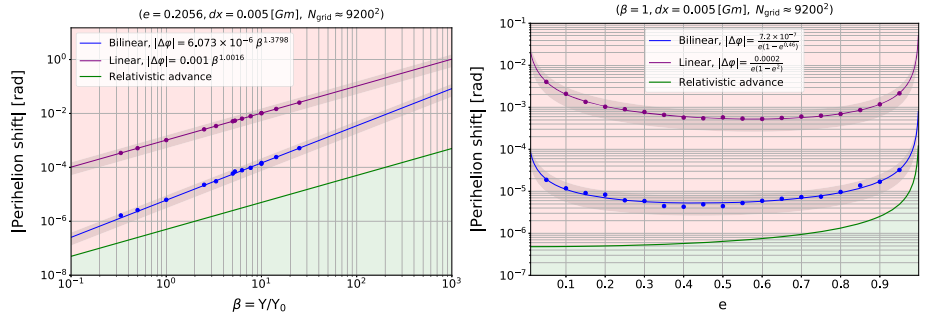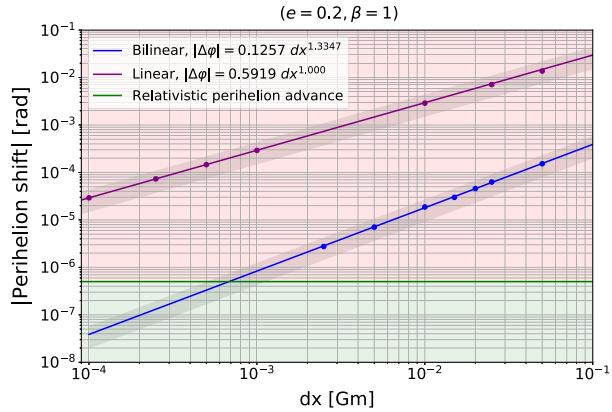
In the case of the linear interpolation, the discontinuity leads to an effective advance of the perihelion, which furthermore depends strongly on the initial angle θ. Qualitatively, this can be understood by the angles at which the orbit crosses the discontinuities. Using otherwise the same parameters as above, the results are summarized in Fig. 7. The advance A of the perihelion follows closely a cosine (with a phase shift) $A/dx \sim 0.145 \cos(\theta + 2.34) \sim 0.145 \cos(\theta + 3\pi/4)$. We also checked that the advance of the perihelion changes sign if the initial velocity changes sign. Also note that the average of the advance of the perihelion is close to zero. To generalize the results to include different orbits in N-body simulations especially the ones with high eccentricity and high relativistic parameter, we studied how the perihelion shift depends on eccentricity and relativistic parameter. These are shown in Fig. 7. Given the number of particles considered in current N-body simulations (e.g., 7000^3 in Yu et al. (2017)) and the restrictions of current hardware (same number of lattice points), we see that relativistic corrections of the orbits cannot be detected.

An interesting effect of the discretization for fixed dx is the dependence on eccentricity e. We observed that the deviations scale about as $\frac{1}{e(1-e^2)}$ for the linear interpolation. This means that the effect is largest at extreme values of e. The deviation is also proportional to Υ, while the relativistic correction is proportional to $\frac{\Upsilon}{1-e^2}$.

The code for such tests and for other parameters can be obtained from the authors.

---

[7] For example, after 3 turns, we divide the total angle by 3, and we do not take the difference between the angle for 3 and 2 turns. Of course, errors on these points will average out somewhat.

Fig. 6 Dependence of the standard deviation of the perihelion shift for the two methods, as a function of dx. For the bilinear method, numerical fluctuations are too large to get reliable results for $dx \lesssim 10^{-3}$. The gray regions around the fitted blue and magenta lines show the 50% deviation from the central fit value

$(e = 0.2, \beta = 1)$

Bilinear, $|\Delta\varphi| = 0.1257 \, dx^{1.3347}$
Linear, $|\Delta\varphi| = 0.5919 \, dx^{1.000}$
Relativistic perihelion advance

$(e = 0.2056, dx = 0.005 \, [Gm], \, N_{\text{grid}} \approx 9200^2)$

Bilinear, $|\Delta\varphi| = 6.073 \times 10^{-6} \, \beta^{1.3798}$
Linear, $|\Delta\varphi| = 0.001 \, \beta^{1.0016}$
Relativistic advance

$(\beta = 1, dx = 0.005 \, [Gm], \, N_{\text{grid}} \approx 9200^2)$

Bilinear, $|\Delta\varphi| = \frac{7.2 \times 10^{-7}}{e(1 - e^{0.01})}$
Linear, $|\Delta\varphi| = \frac{0.0002}{e(1 - e^2)}$
Relativistic advance

Fig. 7 Comparison of linear versus bilinear interpolation. Left—behavior as a function of the theoretical relativistic parameter β. The green line shows the relativistic perihelion advance, and therefore the green/magenta area determines the regions for β where one cannot detect the perihelion advance. Note that for the parameters e, dx, and N_{grid} used in the figure, no method is able to detect relativistic corrections. Right—the same study, now as a function of eccentricity e. According to the green/magenta regions, the relativistic effects cannot be detected for the choice of β, dx, and N_{grid}

6 Conclusions

We see that the numerical study of relativistic effects can have two problems. First, the integration method must choose a small enough time step to reach a precision which is better than the size of the relativistic correction. Second, if, additionally, the forces are discretized, the grid size must be quite fine, so that the relativistic corrections are not washed out by the approximation.

In particular, our results allow one to estimate for which choices of Υ, e, and θ, the relativistic effects are larger than the numerical and discretization effects generated by h and dx.

A Newtonian and relativistic orbits

Here, first we review the equations of motion and properties of the orbits in the mechanics of Newtonian particles, and then we derive the equations for general relativistic motion. In two-dimensional polar coordinates (r, φ), the Newton equations take the form

$$\ddot{r} - r\dot{\varphi}^2 = -\frac{GM}{r^2}.$$

Writing the angular momentum per mass l in polar coordinates results in

$$l = r^2\dot{\varphi}.$$

Changing the variable to $u(\varphi) = 1/r(\varphi)$ gives

$$\frac{\mathrm{d}^2}{\mathrm{d}\varphi^2}u + u = \frac{GM}{l^2}. \tag{A.1}$$

To obtain the relativistic perihelion advance, we repeat, for the convenience of the reader, some parts of Stephani (1982, p. 193). The Schwarzschild metric as a spherically symmetric vacuum solution reads,

$$ds^2 = -\left(1 - \frac{r_{\mathrm{sch}}}{r}\right)c^2\mathrm{d}t^2 + \left(1 - \frac{r_{\mathrm{sch}}}{r}\right)^{-1}dr^2 + r^2\left(d\theta^2 + \sin^2\theta\,d\varphi^2\right),$$

where r_{sch} is the Schwarzschild radius, defined by $r_{\mathrm{sch}} = \frac{2GM}{c^2}$, and t and r, θ, φ are, respectively, time and spatial spherical coordinates.[8] To obtain geodesic equations, one starts from the classical action of massive test particle,

$$\mathcal{A} = -m_o \int \sqrt{-g_{\mu\nu}\frac{\mathrm{d}x^\mu}{\mathrm{d}\tau}\frac{\mathrm{d}x^\nu}{\mathrm{d}\tau}}d\tau,$$

where m_o is the mass of the object. Applying Euler–Lagrange equation to the Lagrangian of the test particle gives four equations, in which one sees that the angular momentum is conserved. Therefore, the motion is in a plane. By simple algebra on the equations, one finds the equation of motion as

$$\frac{\mathrm{d}^2}{\mathrm{d}\varphi^2}u + u = \frac{GM}{l^2} + \frac{3}{2}r_{\mathrm{sch}}u^2, \tag{A.2}$$

which is like Newton's equation (A.1) plus a term which is coming from relativistic correction. We solve (A.2) to obtain the relativistic perihelion advance per period, which is well approximated by

$$\Delta\varphi_p \approx \frac{3\pi r_{\mathrm{sch}}}{a\left(1 - e^2\right)}, \tag{A.3}$$

where a is semi-major axis and e is the eccentricity of the orbit.

For Mercury, this leads to the well-known perihelion advance of 42.98 arc sec per century, or ~ 0.103 arc sec per period.

---

[8] In this section, θ is not the angle of the major axis, but just one of the three Euler coordinates.

B The parameterization of orbits

To see the effect of discretization on different orbits in N-body simulations, we parameterize a general orbit with three parameters (Υ, θ, e), where e is the eccentricity, Υ is the relativistic parameter at perihelion, and θ is the angle of semi-major axis with the lattice squares. It is important to note that these parameters are enough to explain any closed orbits in N-body simulations. Moreover, having the three parameters one could uniquely construct the mass of central object as well as the initial position and velocity of the particle.

Relativistic parameter The relativistic parameter $\Upsilon = \frac{r_{\text{sch}}}{r_{\text{per}}}$, for a fixed mass of central object it shows the scale of the orbits and for a fixed size of the orbit it is an indicator of the mass of central object. If we assume that the mass of central object is fixed, by changing the relativistic parameter, different quantities of the orbit would scale as follows:

$$
M \rightarrow M, \qquad r_{\text{per}} \rightarrow \frac{r_{\text{per}}}{\Upsilon},
$$
$$
T \rightarrow \Upsilon^{3/2} T, \qquad v_{\text{per}} \rightarrow \sqrt{\Upsilon} v_{\text{per}}. \tag{B.1}
$$

r_{per} is the perihelion radius, T is the period of the orbit, and v_{per} is the velocity of the object in the perihelion point. To rescale the orbit for the fixed central body mass and fixed eccentricity, one has to change the initial conditions as follows to obtain the new orbit:

$$
x_0 = \frac{r_{\text{per}}}{\Upsilon}, \qquad y_0 = 0,
$$
$$
v_x = 0, \qquad v_y = \sqrt{\Upsilon} v_{\text{per}}.
$$

We could of course change the central object mass instead of changing the size of the orbit while having the same relativistic parameter.

Eccentricity Another parameter which is important in characterizing an orbit is the eccentricity, to change the eccentricity we keep the semi-major axis length fixed and we change the positions and velocities in the perihelion point to recover the desired eccentricity for the orbits

$$
r_{\text{per}} \rightarrow r_{\text{per}} \frac{1-e}{1-e_0}, \qquad v_{\text{per}} \rightarrow v_{\text{per}} \sqrt{\frac{1+e}{1+e_0}}, \tag{B.2}
$$

where e is the new eccentricity and e_0 is the reference eccentricity (in our case mercury). Note that changing eccentricity also results in changing the perihelion distance and relativistic parameter.

Rotation It appears that the angle between the semi-major axis and the lattice squares is an important parameter specially in the linear force interpolation. To rotate the orbit by angle θ, we can follow the coordinate transformations and start from the following initial condition to obtain the correct orbit,

$$
x_0 = r_{\text{per}} \cos(\theta), \qquad y_0 = r_{\text{per}} \sin(\theta),
$$
$$
v_x = -v_{\text{per}} \sin(\theta), \qquad v_y = v_{\text{per}} \cos(\theta).
$$

In Fig. 8, we illustrate the orbits with different ellipticities, relativistic parameters, and angles obtained from numerical results.

Fig. 8 Some examples of the parameterization of elliptic orbits, which show the role of $\beta = \Upsilon/\Upsilon_0$, e, and θ. The orbits are obtained by solving the differential equations
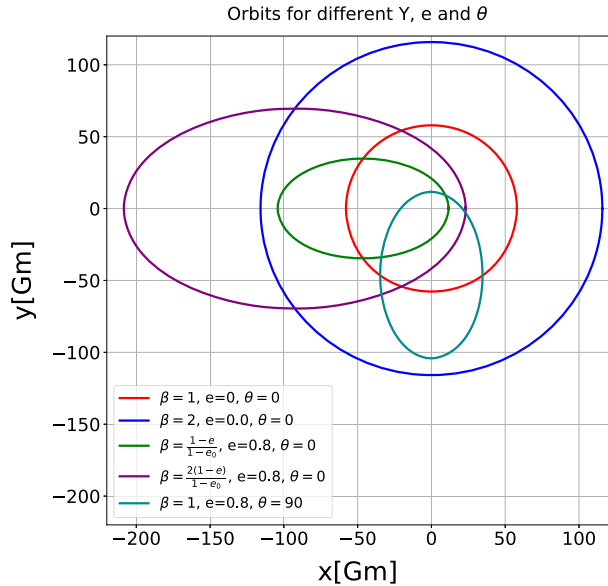
References

Adamek, J., Daverio, D., Durrer, R., Kunz, M.: Gevolution: a cosmological n-body code based on general relativity. J. Cosmol. Astropart. Phys. **2016053**(7), 1–43 (2016a)

Adamek, J., Daverio, D., Durrer, R., Kunz, M.: General relativity and cosmic structure formation. Nat. Phys. **12**, 346 (2016b)

Arnold, D.N.: Differential complexes and numerical stability. In: Tatsien, L. (Ed.) Proceedings of the International Congress of Mathematicians, Vol. I, pp. 137–157. Higher Ed. Press, Beijing (2002)

Back, A.: Runge–Kutta Behavior in the Presence of a Jump Discontinuity. (Preprint) (2005)

Barrera-Hinojosa, C., Li, B.: GRAMSES: a new route to general relativistic N-body simulations in cosmology—I. Methodology and code description. arXiv:1905.08890 [astro-ph.CO]

Butcher, J.C.: Coefficients for the study of Runge-Kutta integration processes. J. Aust. Math. Soc. **3**(2), 185–201 (1963)

Hairer, E., Nørsett, S.P., Wanner, G.: Solving ordinary differential equations. I. Nonstiff problems, volume 8 of Springer Series in Computational Mathematics, 2nd edn. Springer, Berlin (1993)

Hairer, E., Lubich, C., Wanner, G.: Geometric numerical integration illustrated by the Störmer-Verlet method. Acta Numer. **12**, 399–450 (2003)

Hairer, E., Lubich, C., Wanner, G.: Geometric Numerical Integration, volume 31 of Springer Series in Computational Mathematics. Springer, Heidelberg (2010). Structure-Preserving Algorithms for Ordinary Differential Equations, Reprint of the second edition (2006)

Parsa, M., Eckart, A., Shahzamanian, B., Karas, V., Zajacek, M., Zensus, J., et al.: Investigating the relativistic motion of the stars near the supermassive black hole in the galactic center. Astrophys. J. **419**(845), 22 (2017)

Preto, M., Saha, P.: On post-Newtonian orbits and the Galactic-center stars. Astrophys. J. **703**, 1743 (2009)

Springel, V.: The cosmological simulation code gadget-2. Mon. Not. R. Astron. Soc. **364**(4), 1105–1134 (2005)

Stephani, H.: Relativity: An Introduction to Special and General Relativity. Cambridge University Press (Transl. from German By M. Pollock and J. Stewart) (1982)

Teyssier, R.: Cosmological hydrodynamics with adaptive mesh refinement: a new high resolution code called ramses. Astron. Astrophys. **385**, 337–364 (2002)

Yu, H., Emberson, J., Inman, D., et al.: Differential neutrino condensation onto cosmic structure. Nature Astronomy **1**(143), (2017)