Numerical Instability in Gravitational N-Body Simulations: A Comparative Analysis

Kamyar Modjtahedzadeh

Lockheed Martin & Northrop Grumman

November 3, 2025

Abstract

This paper compares Euler and velocity Verlet integration methods for simulating gravitational N-body dynamics using 100 unit-mass particles initially arranged on a 2D square perimeter. The first-order Euler method exhibits catastrophic energy injection leading to unphysical particle ejection when timesteps are too large. The second-order symplectic velocity Verlet algorithm maintains bounded energy errors and preserves phase space structure over extended simulations. At $t = 0.1$ (10,000 iterations, $\Delta t = 10^{-5}$), Verlet maintains organized orbital structure while Euler produces chaotic collapse, demonstrating the critical importance of symplectic integration for conservative Hamiltonian systems. Taylor series analysis shows Verlet achieves $\mathcal{O}(\Delta t^2)$ global accuracy compared to Euler's $\mathcal{O}(\Delta t)$, though system stiffness limits practical timestep advantages for strongly coupled systems ($G \geq 50$).

. . . more to come, in development. . .

1 Simulation and Results of Euler Integration

A square perimeter subject to the gravitational force between its own constructing point particles each with unit mass will be simulated over time using the Euler method. The structure will will start with 100 evenly space particles with the first particle located at $(x,y) = (0,0)$ and the length of each side of the square is 1.

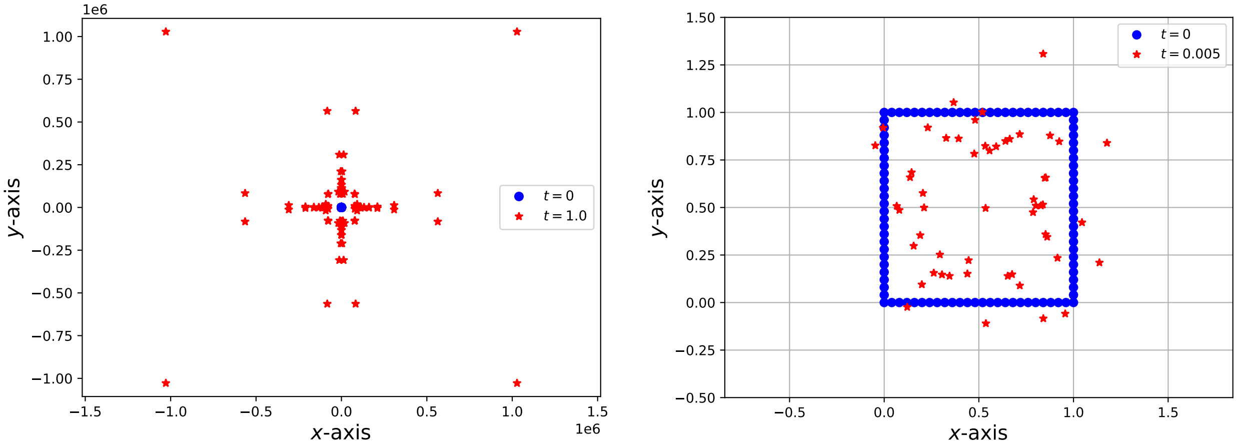
1.1 Parameters and Expectations

Using the Euler method, both simulations model pairwise gravitational interactions through Newton's law of universal gravitation:

$$\vec{F}_{\text{grav}} = -G \cdot \frac{m_1 m_2}{r^2} \hat{r}, \tag{1.1}$$

where G represents the gravitational field strength and is *not* a constant of simulation, albeit a constant of nature. The negative sign indicates that gravity is an attractive force.

Starting at $t_{\text{initial}} = 0$, the first simulation employs a very strong gravitational coupling of $G = 10^6$, timestep $\Delta t = 10^{-3}$, and runs for 10^3 steps spanning a total duration of $t_{\text{final}} = 1$. The second simulation drastically reduces the gravitational constant to $G = 10^2$ while also drastically reducing the timestep length to 5×10^{-6}, thus capturing the system at an earlier moment $t_{\text{final}} = 5 \times 10^{-3}$. Both systems should exhibit near-identical behavior: The point particles should accelerate inward and converge toward the structure's center of mass, forming a compact cluster. The conservation of energy in this isolated gravitational system dictates that particles initially at rest cannot acquire sufficient kinetic energy to overcome the gravitational potential well and escape.



(a) $G = 10^6$, $\Delta t = 10^{-3}$

(b) $G = 10^2$, $\Delta t = 5 \times 10^{-6}$

Figure 1.1: The Euler integration method applied to the same 2D shallow square of 100 particles over 1000 timesteps. When Δt is *not* small enough, as in Figure 1.1a, the structure flies apart, something that is not expected under the attractive force in Equation (1.1). When Δt *is* small enough, as in Figure 1.1b, the system acts as expected; however, over the same number of timesteps (i.e., iterations), a significantly smaller t_{final} is reached.

1.2 Observations and Discrepancies

While a gravitational collapse may have occurred for a seemingly insubstantial fraction of the number of particles in the first simulation, the vast majority of particles have clearly moved far away from the center of mass of the structure's initial configuration, as shown in Figure 1.1a. A direct contradiction of the principle of energy conservation. On the other hand, operating under the same laws of physics, the second simulation, captured at the earlier time of $t = 0.005$ in Figure 1.1b, behaves as expected. The square structure has contracted substantially, with particles migrating inward toward the center of mass. Although a few particles have moved farther than expected, the overall picture depicts a collapsing structure consistent with gravitational theory.

2 Systematic Errors of the Euler Method

2.1 Numerical Error Accumulation

The difference between expected and observed behavior in the first simulation is due to systematic errors introduced by the Euler method.[1] The Euler integration method represents a basic numerical approach for solving ordinary differential equations, updating particle velocities and positions according to the following equations:

$$\vec{a}_i(t) = \vec{F}_i(t)/m_i \tag{2.1}$$

$$\vec{v}_i(t+\Delta t) = \vec{v}_i(t) + \vec{a}_i(t)\Delta t \tag{2.2}$$

$$\vec{r}_i(t+\Delta t) = \vec{r}_i(t) + \vec{v}_i(t)\Delta t\,. \tag{2.3}$$

The issue with this method is that it treats continuously varying forces as piecewise constant values. At the beginning of each iteration, the method evaluates all forces based on current particle positions, computes accelerations, and then applies these accelerations uniformly throughout the entire duration of the timestep. This approximation fails to account for the fact that particles move *during* the timestep. The gravitational force's inverse-square dependence on distance causes the particles to approach one another, forces increase rapidly, and the assumption of constant acceleration over the timestep becomes increasingly inaccurate in this approximation. The Euler method's approximation errors introduce spurious energy into the system at each timestep. When a particle experiences a strong attractive force and accelerates toward another particle, the Euler method applies that force for the full timestep duration even as the particle moves closer and the actual force should be increasing. This delayed response to strengthening forces causes particles to acquire velocities that are systematically too large, effectively injecting kinetic energy that has no physical origin. One workaround is to use a very short timestep, but that would require tremendously more iterations and computational power to achieve a desired t_{final}. For the gravitational N-body problem, particularly when particles can approach closely and generate stiff dynamics through the inverse-square force law, the Euler integration method will eventually fail regardless of how small the timestep becomes, unless the timestep is reduced to impractically tiny values.

2.2 Symplectic Integration for Energy Conservation

The Euler method is simply inadequate for simulating conservative systems over long periods. The non-symplectic method's first-order accuracy makes it unsuitable for problems requiring energy conservation. One solution for such problems could be to adopt symplectic, higher-order integration schemes specifically designed for Hamiltonian systems. A symplectic integrator is a numerical method that accurately conserves important physical quantities over sufficiently large time scales. A crucial and necessary condition for standard, explicit integrators to work correctly on a specific class of conservative dynamical systems is *time reversibility*, something which the standard explicit Euler method is not (see Appendix A.1).

The **velocity Verlet** algorithm, for instance, is a second-order symplectic integrator that exhibits excellent long-term energy conservation properties. Symplectic integrators preserve the geometric structure of phase space and maintain bounded energy errors even over extremely long simulation times. Time reversibility is not the only condition for a method to be symplectic, the defining property is the preservation of the symplectic structure. A numerical method is defined as symplectic if its area in phase space is preserved.

3 Velocity Verlet Integration

The velocity Verlet algorithm is a second-order symplectic integrator designed to preserve energy and phase space structure in Hamiltonian systems. In four stages per timestep, the method advances a particle system through discrete timesteps while maintaining time reversibility and symplectic structure.

1. The current acceleration is computed from the force acting on the particle: $\vec{a}(t) = \vec{F}(t)/m$.

[1] **First-Order Accuracy:** The Euler method is called a first-order method because the error per step is proportional to the square of the step size (Δt^2), and the global error is proportional to the step size (Δt). This means that if the step size is halved, the error is also approximately halved. The method is linear in its error reduction with respect to the step size.

2. The position is advanced using the current velocity and acceleration:

$$\vec{r}(t+\Delta t) = \vec{r}(t) + \vec{v}(t)\Delta t + \frac{1}{2}\vec{a}(t)\Delta t^2 . \tag{3.1}$$

3. Forces are recomputed at the new particle positions to obtain $\vec{F}(t+\Delta t)$, from which the new acceleration follows: $\vec{a}(t+\Delta t) = \vec{F}(t+\Delta t)/m$.
4. The velocity is updated using the average of the old and new accelerations:

$$\vec{v}(t+\Delta t) = \vec{v}(t) + \frac{1}{2}\left[\vec{a}(t) + \vec{a}(t+\Delta t)\right]\Delta t . \tag{3.2}$$

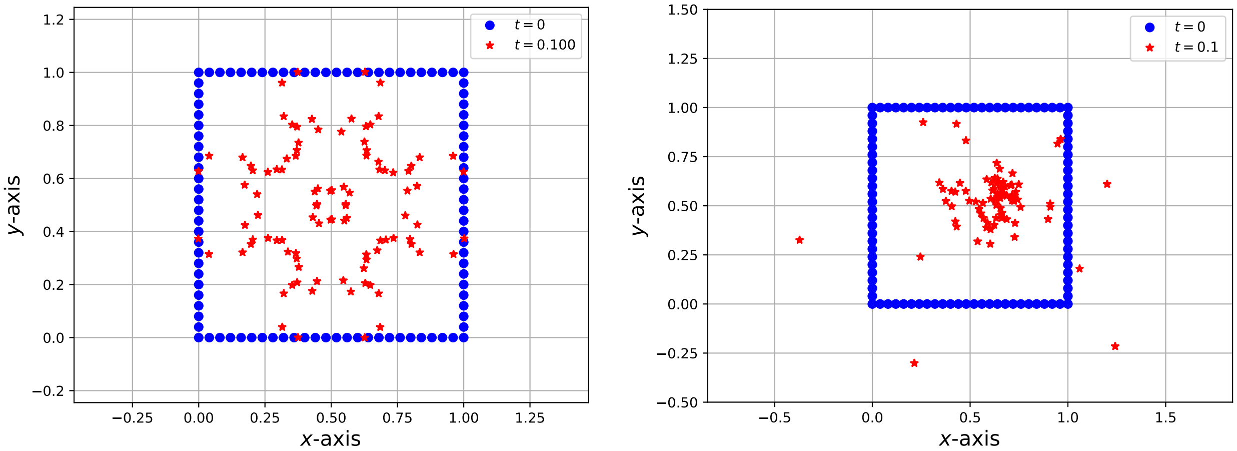
The velocity Verlet algorithm is a second-order method with local truncation error proportional to Δt^3 and global error proportional to Δt^2. This represents a significant improvement over the first-order Euler method, which has local error proportional to Δt^2 and global error proportional to Δt. (See Appendix B for more details.)

3.1 Performance Analysis: Stiff Gravitational Systems

The modified gravitational force between particles i and j is given by,

$$\vec{F}'_{\text{grav}} = -G \cdot \frac{m_i m_j}{r_{ij}^2 + \varepsilon^2}\hat{r}_{ij} \tag{3.3}$$

where ε is a softening parameter that prevents numerical divergence at small separations. For strongly coupled gravitational systems,[2] testing reveals that over a long enough time the velocity Verlet method will preserve an organized structure while the Euler method produces a dispersed cluster collapsed to center, as seen in Figure 3.1. Physically, this means that Verlet integration better preserves the system's dynamical invariants. The method has bounds to the energy error making particles follow more realistic orbits, some particles remain in quasi-periodic trajectories developing persisting organized patterns. Euler integration on the other hand injects artificial energy into the system causing the particles to gain excessive kinetic energy and ultimately escape to infinity or collapse too aggressively resulting in a loss of structure. The "organized structure" is particles maintaining correlated motion (orbits, oscillations around center, etc.) rather than random scattering. This preservation of geometric structure allows the velocity Verlet integration to maintain a physically realistic so-called final configuration over long timescales whereas the Euler method's energy drift dominates the dynamics.



(a) Velocity Verlet method

(b) Euler integration method

Figure 3.1: Gravitational collapse of a square perimeter made up of 100 unit-mass point particles using velocity Verlet and Euler integration methods. Both methods run over 10^4 iterations with $\Delta t = 10^{-5}$, $G = 10$, and $\varepsilon = 0.05$. It is evident that the velocity Verlet integration has preserved phase space related quantities as it has maintained a structure. Figure 3.1b however is chaotic and indicates no preservation of such quantities.

[2] I.e., $G \geq 50$

Appendix A: Time Reversibility of Certain Integration Methods

The laws of motion for conservative physical systems are time-reversible. If the system is ran forward in time such that $t_{\ell+1} = t_\ell + \Delta t$, and then ran backward using the same exact method and timestep, the ending should match the start. Time-irreversibility causes phase space area to drift away from its true value in an uncontrolled manner; in turn, certain physical properties may not be preserved by the instable simulation.

A.1 Explicit Euler Method (irreversible)

The explicit Euler method advances the state based on standard kinematic equations of motion;

$$x_{\ell+1} = x_\ell + v_\ell \Delta t \tag{A.1}$$

$$v_{\ell+1} = v_\ell + a_\ell \Delta t \,. \tag{A.2}$$

The above formulas step forward from t_ℓ to $t_{\ell+1}$ which are Δt apart. Rearrange them to go backward in time from $t_{\ell+1}$ to t_ℓ;

$$x_\ell = x_{\ell+1} - v_\ell \Delta t \tag{A.3}$$

$$v_\ell = v_{\ell+1} - a_\ell \Delta t \,. \tag{A.4}$$

So when at $t_{\ell+1}$, to know the state at time t_ℓ, v_ℓ and a_ℓ must already be known, which goes against the very reason it was started.

A.2 Basic Størmer–Verlet Algorithm (reversible)

The Verlet algorithm is a second-order method that relies on the state at the current time and the state of the time point that is one timestep behind t_ℓ to find the state at $t_{\ell+1}$. The second-order differential equation is:

$$x_{\ell+1} = 2x_\ell - x_{\ell-1} + a_\ell \Delta t^2 \,. \tag{A.5}$$

Starting at $t_{\ell+1}$ and going backward to $t_{\ell-1}$,

$$x_{\ell-1} = 2x_\ell - x_{\ell+1} + a_\ell \Delta t^2 \,, \tag{A.6}$$

which is possible since the state at one time point before $t_{\ell+1}$ is known in the Størmer method.

A.3 Classic RungeKutta Method (irreversible)

The classic RungeKutta method (RK4) is a fourth-order method that can solve a wide range of initial value ordinary differential equations without being tailored to a specific type of problem. To calculate $x_{\ell+1}$:

$$x_{\ell+1} = x_\ell + \frac{\Delta t}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \,, \tag{A.7}$$

where the k values are the intermediate steps whose quantities depend on $f(t,x) = \dot{x}$ such that:

$$k_1 = f\left(t_\ell, x_\ell\right) \tag{A.8}$$

$$k_2 = f\left(t_\ell + \frac{\Delta t}{2}, x_\ell + \frac{k_1 \Delta t}{2}\right) \tag{A.9}$$

$$k_3 = f\left(t_\ell + \frac{\Delta t}{2}, x_\ell + \frac{k_2 \Delta t}{2}\right) \tag{A.10}$$

$$k_4 = f\left(t_\ell + \Delta t, x_\ell + k_3 \Delta t\right) \,. \tag{A.11}$$

This structure of k means that there is no standard formula that simply reverses the forward step. This asymmetry is ultimately the reason for drift. RK4 methods offer improved accuracy through multiple force evaluations per timestep, though they are not symplectic and thus do not guarantee the same level of energy stability as Verlet-family integrators.

Appendix B: Truncation Error Analysis via Taylor Expansion

The Taylor series expansion of $\vec{r}(t+\Delta t)$ about t is,

$$\vec{r}_{\text{exact}}(t+\Delta t) = \vec{r}(t) + \dot{\vec{r}}(t)\Delta t + \frac{1}{2}\ddot{\vec{r}}(t)\Delta t^2 + \frac{1}{6}\dddot{\vec{r}}(t)\Delta t^3 + \mathcal{O}(\Delta t^4)\,. \tag{B.1}$$

The Euler approximation is,

$$\vec{r}_{\text{Euler}}(t+\Delta t) = \vec{r}(t) + \vec{v}(t)\Delta t\,, \tag{B.2}$$

so the leading error term is $\frac{1}{2}\ddot{\vec{r}}(t)\Delta t^2$; therefore, the local error is $\mathcal{O}(\Delta t^2)$. Similarly, the Verlet approximation is,

$$\vec{r}_{\text{Verlet}}(t+\Delta t) = \vec{r}(t) + \vec{v}(t)\Delta t + \frac{1}{2}\vec{a}(t)\Delta t^2\,, \tag{B.3}$$

with the leading error and local error being $\frac{1}{6}\dddot{\vec{r}}(t)\Delta t^3$ and $\mathcal{O}(\Delta t^3)$ respectively. Moreover, global errors are the accumulation of local errors over the time of simulation: $n = t_{\text{final}}/\Delta t$, where n would be the number of timesteps iterated by the simulation. For the Euler method: $n \times \mathcal{O}(\Delta t^2) = (t_{\text{final}}/\Delta t) \times \mathcal{O}(\Delta t^2) \to \mathcal{O}(\Delta t)$. Similarly, for Verlet integration: $n \times \mathcal{O}(\Delta t^3) = (t_{\text{final}}/\Delta t) \times \mathcal{O}(\Delta t^3) \to \mathcal{O}(\Delta t^2)$.