OrbitLab: An Educational N-body Sandbox for Comparing Numerical Integrators

Slava Savelyev

Independent student researcher, Kursk, Russia

2025-09-17 (preprint / technical report)

Abstract

Numerical integration choices strongly affect stability and physical fidelity in N-body simulations. This technical report presents OrbitLab, an open-source Python sandbox for educational N-body gravity (O(N^2) baseline) with a focus on reproducibility and explainability. We implement three common integrators—explicit Euler, classical fourth-order Runge–Kutta (RK4), and a symplectic leapfrog scheme—and compare them on a simple two-body near-circular orbit ("Sun–Earth" in dimensionless units). For 5,000 steps at dt = 0.002, Euler exhibits noticeable energy drift (≈5.6×10^{-8}, ≈3.7% relative to |E(0)|), while leapfrog and RK4 keep drift near machine precision (≈3.0×10^{-17} and ≈2.0×10^{-20}). On the same workload, RK4 is ~2.3× slower than Euler, illustrating a practical accuracy–performance tradeoff. The project includes tests, a CLI, and a one-command demo that outputs trajectory and energy-drift plots.

Keywords: N-body simulation, numerical integration, symplectic methods, energy drift, reproducibility

1. Introduction

Educational physics simulations often prioritize visual output over numerical quality. However, even simple choices like the time integrator can change the qualitative behavior of a dynamical system (e.g., spiraling orbits caused by energy drift). OrbitLab is designed as a compact, readable, and reproducible reference implementation for experimenting with integration schemes in gravitational N-body dynamics.

2. Model and Notation

We consider an N-body system in D dimensions with masses m_i and positions x_i(t). The acceleration of body i is computed as the sum of pairwise Newtonian interactions:

a_i(t) = G * Σ_{j≠i} m_j * (x_j - x_i) / (||x_j - x_i||^2 + ε^2)^(3/2).

OrbitLab uses dimensionless units with a default gravitational constant G = 1.0 and a small softening ε > 0 to avoid singularities when distances become very small.

3. Integrators Implemented

OrbitLab includes three time-stepping schemes. In all cases, we step positions x and velocities v with a fixed time step dt.

3.1 Explicit Euler

Euler updates velocity and position using the current acceleration a(x(t)):
v(t+dt) = v(t) + dt * a(x(t)),
x(t+dt) = x(t) + dt * v(t).
It is simple but often unstable for orbital problems.

3.2 Classical Runge–Kutta (RK4)

RK4 evaluates the right-hand side at several intermediate points within the step to achieve fourth-order accuracy. It can be very accurate for smooth problems but is computationally more expensive (multiple acceleration evaluations per step).

3.3 Symplectic Leapfrog (Velocity-Verlet)

Leapfrog is a symplectic integrator, which tends to preserve geometric structure in Hamiltonian systems and often shows bounded energy error over long times. In OrbitLab we use a standard velocity-Verlet style update.

4. Experimental Setup

Scenario: a two-body near-circular orbit with m_sun = 1.0, m_earth = 3×10^{-6}, initial distance 1.0, and v_earth ≈ 1.0. The Sun's initial velocity is set to keep total momentum near zero.

We simulate for 5,000 steps with dt = 0.002 and compute:

- Total energy E(t) = kinetic + potential (with the same softening used in acceleration).
- Energy drift ΔE = E(T) − E(0) and relative drift |ΔE| / |E(0)|.
- Wall-clock runtime for the full simulation (machine-dependent; used only for rough comparison).

5. Results

Integrator	dt	steps	|ΔE|	|ΔE|/|E(0)|	runtime (s)
euler	0.002	5000	5.575e-08	3.717e-02	0.607
leapfrog	0.002	5000	3.035e-17	2.023e-11	0.671
rk4	0.002	5000	1.969e-20	1.313e-14	1.406

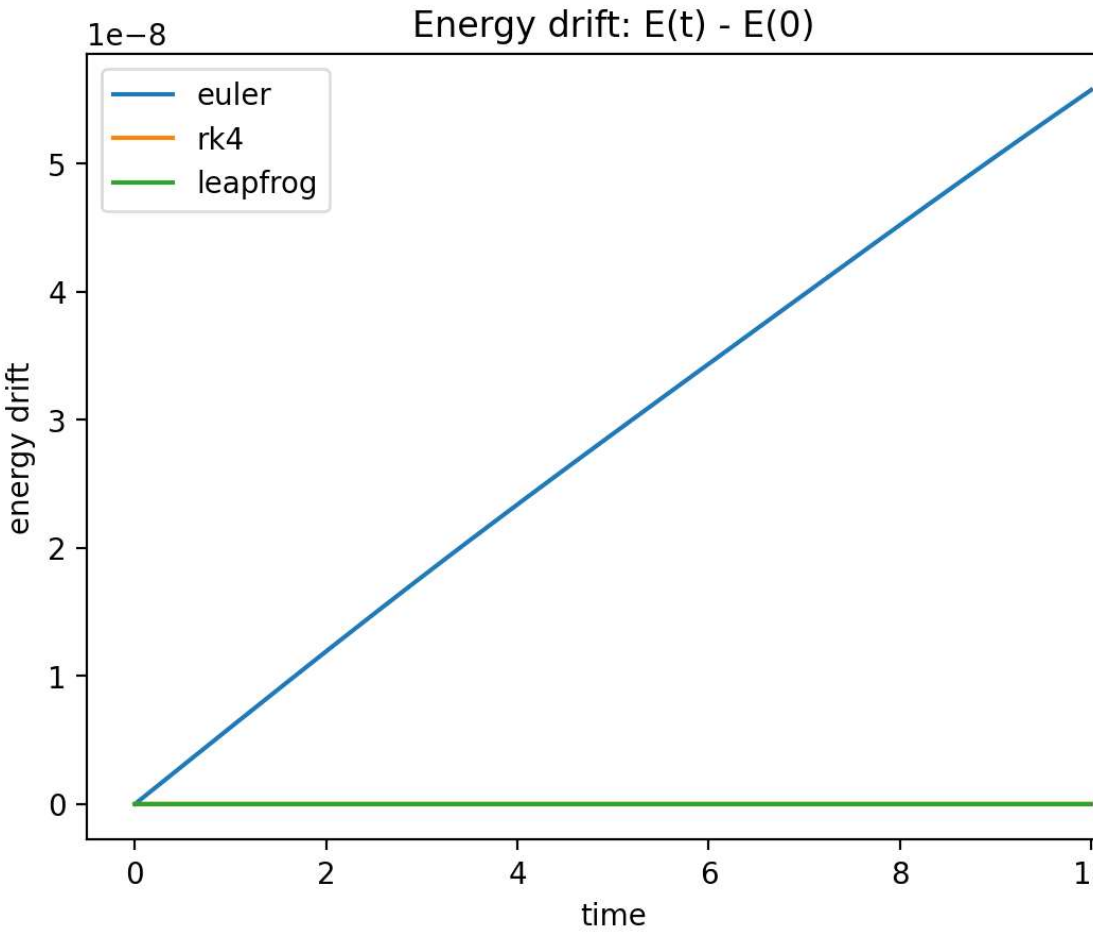

Figure 1. Energy drift over time for Euler, leapfrog, and RK4 (Sun–Earth scenario).

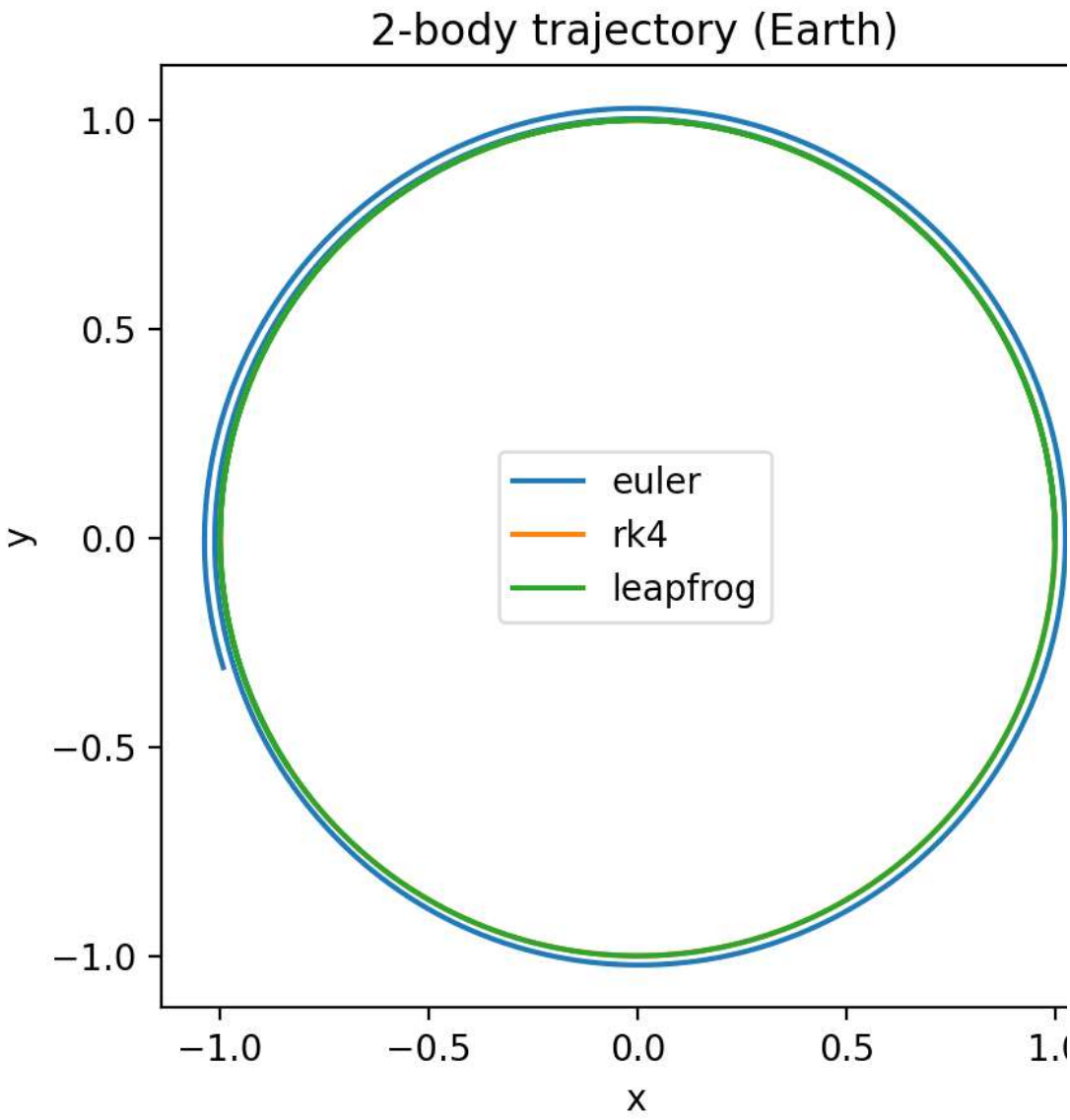

Figure 2. Trajectory plot for the 2-body orbit (Sun–Earth scenario).

6. Discussion

Even on a small two-body example, explicit Euler introduces visible energy drift, which can qualitatively change orbital behavior. Leapfrog reduces drift dramatically at roughly the same runtime cost as Euler, while RK4 achieves extremely small drift at a higher cost due to multiple force evaluations per step. These results highlight that "accuracy" has multiple dimensions: local error order, long-term stability, and computational cost.

7. Limitations and Future Work

- OrbitLab intentionally uses a simple O(N^2) gravity kernel; faster methods (Barnes–Hut, particle-mesh, GPU kernels) are future work.
- The default demo uses dimensionless units and a toy Sun–Earth setup; more realistic unit systems can be added.
- Long-horizon benchmarks (millions of steps) would better demonstrate bounded energy error for symplectic schemes.

8. Reproducibility

Code and instructions are provided in the OrbitLab repository. To reproduce the demo plots:

- Create a virtual environment (Python 3.12/3.13 recommended on Windows).
- Install dependencies: pip install -r requirements.txt
- Run: python -m orbitlab.demo
- Outputs will be saved into outputs/ (trajectory.png, energy_drift.png, summary.json).

References

E. Hairer, C. Lubich, and G. Wanner, Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations, 2nd ed., Springer, 2006.

W. H. Press et al., Numerical Recipes: The Art of Scientific Computing, 3rd ed., Cambridge University Press, 2007.