Oscillations of a Driven Quadruple Pendulum

Parth Bhargava (A0310667E) · Soham Bhar (A0311156R)

PC3236 Computational Methods in Physics

April 08, 2026

1 Abstract

The chaotic dynamics of a driven, damped quadruple pendulum are studied numerically. Four equal point masses on rigid, massless rods are coupled through the Lagrangian, driven by a periodic torque $\tau(t) = \tau_0 \cos(\omega_d t)$ at the first pivot, and dissipated by absolute environmental viscous damping $-b\dot{\theta}_i$ on each pendulum. The resulting four coupled second-order ODEs are recast as eight first-order equations and integrated with a hand-coded fourth-order Runge-Kutta (RK4) scheme at $\Delta t = 5 \times 10^{-4}$ s. At each RK4 stage the 4×4 mass-matrix system $\boldsymbol{M\alpha} = \boldsymbol{F}$ is solved via LU decomposition (`np.linalg.solve`). With $\tau_0 = 8.0$ N · m, $\omega_d = 1.6\pi$ rad/s, and $b = 0.10$ N · m · s/rad, all four pendulums undergo aperiodic, bounded oscillations and the outermost pendulum completes full rotations, confirming the system is in a genuinely chaotic regime. Two nearby trajectories ($\Delta\theta_1(0) = 10^{-6}$ rad) grow apart exponentially over roughly 7 orders of magnitude across the first 20 s of integration, and the bifurcation diagram shows a clear transition from periodic to chaotic response as τ_0 increases past roughly 6 N · m. The custom RK4 agrees with scipy's adaptive RK45 to within 10^{-11} rad for the first few seconds, then diverges rapidly as the two solvers accumulate different truncation errors in the chaotic regime.

2 Description of the Problem

A single pendulum swinging under gravity is among the cleanest examples of periodic motion in classical mechanics. Couple two pendulums end-to-end and the dynamics become richer: the shared pivot transmits energy between the two degrees of freedom, and for large enough swings the system can become chaotic. Each additional pendulum in the chain adds two dimensions to the phase space and strengthens the nonlinear coupling, so by the time four pendulums hang in series the eight-dimensional phase space is vast enough that even infinitesimally close initial conditions can lead to exponentially diverging trajectories.

Attaching a periodic torque at the top pivot pumps energy into the chain. Without any damping this energy accumulates without bound, which causes the numerical simulation to blow up within seconds. Adding an absolute viscous drag $-b\dot{\theta}_i$ on each pendulum provides an energy sink, and the interplay between this dissipation and the drive is what generates the rich dynamics we observe: at low driving amplitudes the damping dominates and the pendulums settle into a periodic limit cycle; increase the drive far enough and the system can no longer find a periodic steady state, instead exploring a strange attractor in phase space.

The specific system studied here consists of four point masses $m_1 = m_2 = m_3 = m_4 = 1.0$ kg connected by rigid rods of lengths $L_1 = 0.40$, $L_2 = 0.35$, $L_3 = 0.30$, $L_4 = 0.25$ m. A periodic torque $\tau_0 \cos(\omega_d t)$ with $\tau_0 = 8.0$ N·m and $\omega_d = 1.6\pi \approx 5.03$ rad/s is applied at the pivot of the first pendulum. The driving frequency is close to the small-angle natural frequency of the first pendulum ($\sqrt{g/L_1} \approx 4.95$ rad/s), so the first bob picks up energy from the torque effectively. Each pendulum experiences absolute environmental viscous damping with coefficient $b = 0.10$ N·m·s/rad. The goal is to determine the angular trajectories $\theta_1(t), ..., \theta_4(t)$ and characterise the transition from periodic to chaotic motion as the driving amplitude varies.

3 Equations to be Solved

3.1 Coordinates and Kinematics

Let θ_i be the angle of the i-th rod measured from the downward vertical. The Cartesian positions of the four bobs are:

$$x_1 = L_1 \sin\theta_1, \quad y_1 = -L_1 \cos\theta_1 \tag{1}$$

$$x_2 = x_1 + L_2 \sin\theta_2, \quad y_2 = y_1 - L_2 \cos\theta_2 \tag{2}$$

$$x_3 = x_2 + L_3 \sin\theta_3, \quad y_3 = y_2 - L_3 \cos\theta_3 \tag{3}$$

$$x_4 = x_3 + L_4 \sin\theta_4, \quad y_4 = y_3 - L_4 \cos\theta_4 \tag{4}$$

Each bob inherits the velocity of its parent joint, so the velocity components grow progressively more complicated down the chain.

3.2 Lagrangian and Equations of Motion

The kinetic energy $T = \frac{1}{2}\sum_i m_i(\dot{x}_i^2 + \dot{y}_i^2)$ and potential energy $V = \sum_i m_i g y_i$ give the Lagrangian $\mathcal{L} = T - V$. Applying the Euler-Lagrange equations with generalised forces Q_i on the right-hand side yields a 4×4 matrix equation:

$$\boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} = \boldsymbol{F}\big(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, t\big) \tag{5}$$

The mass matrix $\boldsymbol{M}$ is symmetric and configuration-dependent; the forcing vector $\boldsymbol{F}$ collects the centrifugal coupling, gravitational torques, external drive, and damping.

3.3 Mass Matrix

The general structure of the 4×4 symmetric mass matrix is:

$$M_{ii} = \left(\sum_{k=i}^{4} m_k\right) L_i^2, \qquad M_{ij} = M_{ji} = \left(\sum_{k=\max(i,j)}^{4} m_k\right) L_i L_j \cos\big(\theta_i - \theta_j\big), \quad i \neq j \tag{6}$$

The diagonal elements are constant, while the off-diagonal elements oscillate with the angle differences. Written out, the ten independent elements are Equation 7 through Equation 16:

$$M_{11} = (m_1 + m_2 + m_3 + m_4)L_1^2 \tag{7}$$

$$M_{12} = (m_2 + m_3 + m_4)L_1 L_2 \cos(\theta_1 - \theta_2) \tag{8}$$

$$M_{13} = (m_3 + m_4)L_1 L_3 \cos(\theta_1 - \theta_3) \tag{9}$$

$$M_{14} = m_4 L_1 L_4 \cos(\theta_1 - \theta_4) \tag{10}$$

$$M_{22} = (m_2 + m_3 + m_4)L_2^2 \tag{11}$$

$$M_{23} = (m_3 + m_4)L_2 L_3 \cos(\theta_2 - \theta_3) \tag{12}$$

$$M_{24} = m_4 L_2 L_4 \cos(\theta_2 - \theta_4) \tag{13}$$

$$M_{33} = (m_3 + m_4)L_3^2 \tag{14}$$

$$M_{34} = m_4 L_3 L_4 \cos(\theta_3 - \theta_4) \tag{15}$$

$$M_{44} = m_4 L_4^2 \tag{16}$$

3.4 Forcing Vector

The forcing vector $\boldsymbol{F}$ contains centrifugal coupling, gravity, the external drive, and damping. The general rule for element i is:

$$F_i = -\sum_{j \neq i} C_{ij}\dot{\theta}_j^2 \sin\big(\theta_i - \theta_j\big) - \left(\sum_{k=i}^{4} m_k\right) g L_i \sin\theta_i + Q_i - b\dot{\theta}_i \tag{17}$$

where $C_{ij} = \left(\sum_{k=\max(i,j)}^{4} m_k\right) L_i L_j$ is the coupling coefficient, $Q_1 = \tau_0 \cos(\omega_d t)$, $Q_2 = Q_3 = Q_4 = 0$, and $-b\dot{\theta}_i$ is the viscous damping torque. Written out for F_1:

$$\begin{aligned} F_1 = &-(m_2+m_3+m_4)L_1L_2\dot{\theta}_2^2\sin(\theta_1-\theta_2) - (m_3+m_4)L_1L_3\dot{\theta}_3^2\sin(\theta_1-\theta_3) \\ &-m_4L_1L_4\dot{\theta}_4^2\sin(\theta_1-\theta_4) - (m_1+m_2+m_3+m_4)gL_1\sin\theta_1 \\ &+\tau_0\cos(\omega_d t) - b\dot{\theta}_1 \end{aligned} \tag{18}$$

The remaining rows F_2, F_3, F_4 follow the same structure: $F_i = -\sum_{j\neq i} C_{ij}\dot{\theta}_j^2 \sin\big(\theta_i - \theta_j\big)$ minus gravity, with no external drive and a damping term $-b\dot{\theta}_i$ on each.

3.5 State-Space Formulation

Defining the state vector $\boldsymbol{y} = (\theta_1, \theta_2, \theta_3, \theta_4, \omega_1, \omega_2, \omega_3, \omega_4)^T$ where $\omega_i = \dot{\theta}_i$, the system becomes eight first-order ODEs:

$$\dot{\theta}_i = \omega_i, \qquad \dot{\omega}_i = \big[\boldsymbol{M}^{-1}\boldsymbol{F}\big]_i, \quad i = 1,2,3,4 \tag{19}$$

3.6 Fourth-Order Runge-Kutta Method

Given the ODE $\dot{\boldsymbol{y}} = \boldsymbol{f}(t, \boldsymbol{y})$, a single RK4 step from t_n to $t_{n+1} = t_n + h$ is:

$$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n) \tag{20}$$

$$\boldsymbol{k}_2 = \boldsymbol{f}\left(t_n + \frac{h}{2}, \boldsymbol{y}_n + \frac{h}{2}\boldsymbol{k}_1\right) \tag{21}$$

$$\boldsymbol{k}_3 = \boldsymbol{f}\left(t_n + \frac{h}{2}, \boldsymbol{y}_n + \frac{h}{2}\boldsymbol{k}_2\right) \tag{22}$$

$$\boldsymbol{k}_4 = \boldsymbol{f}(t_n + h, \boldsymbol{y}_n + h\boldsymbol{k}_3) \tag{23}$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \frac{h}{6}(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4) \tag{24}$$

The local truncation error is $O(h^5)$, giving global error $O(h^4)$.

4 Description of Computational Methodology and Implementation

4.1 Overview

The simulation is implemented in a single Python script (`quadruple_pendulum.py`, 491 lines). The code separates into four layers:

1. **Physical parameters** ($g, L_i, m_i, \tau_0, \omega_d, b$) and time-grid constants at module scope (lines 48–58).
2. **Physics functions**: `mass_matrix(θ)` and `forcing(θ, ω, t)` return NumPy arrays implementing Equation 6 through Equation 18 (lines 63–117).
3. **Integrator**: `derivs(t, y)` assembles the right-hand side of Equation 19 by solving $\boldsymbol{M}\boldsymbol{\alpha} = \boldsymbol{F}$ with `np.linalg.solve`, and `rk4_step` / `integrate_rk4` advance the state vector (lines 120–150).
4. **Diagnostics**: seven plotting routines, an animation generator, and a scipy cross-check (lines 210–441).

4.2 Code-to-Equation Mapping

Code (line numbers)	Corresponding equation / mathematical operation
Lines 48–58	Physical constants $g, L_i, m_i, \tau_0, \omega_d, b$ and time-grid parameters.
Lines 63–83 `mass_matrix()`	Mass matrix Equation 6, Equation 7 through Equation 16. Diagonal entries use $\sum_{k\geq i} m_k$; off-diagonal entries use $\cos(\theta_i - \theta_j)$. Lower triangle filled by symmetry.
Lines 86–117 `forcing()`	Forcing vector Equation 17, Equation 18. Each row implements centrifugal coupling, gravity, the drive (row 0 only), and damping $-b\omega_i$.
Lines 120–127 `derivs()`	State-space formulation Equation 19: solves $\boldsymbol{M}\ddot{\boldsymbol{\theta}} = \boldsymbol{F}$ via `np.linalg.solve` (LU decomposition), returns $\left[\omega_i, \ddot{\theta}_i\right]$.
Lines 132–138 `rk4_step()`	RK4 formula Equation 22: four evaluations of `f(t, y)` at intermediate points, combined with weights $1/6, 1/3, 1/3, 1/6$.
Lines 141–150 `integrate_rk4()`	Time-stepping loop: uniform grid via `np.linspace`, advancing the 8-component state vector one RK4 step per iteration.
Lines 155–174 `kinetic_energy()`, `potential_energy()`	Kinetic energy $T = \frac{1}{2}\sum_i m_i(\dot{x}_i^2 + \dot{y}_i^2)$ and potential energy $V = \sum_i m_i g y_i$ from Equation 1 through Equation 4.
Lines 380–441 `animate_pendulum()`	Bob positions from Equation 1 through Equation 4 computed at each frame; nearest-neighbour frame selection by `np.searchsorted` (bisection). `FuncAnimation` updates rods and bobs.

Table 1: Mapping between code sections and the mathematical equations they implement.

4.3 Why These Syntax Choices

- **`np.linalg.solve(M, F)` rather than explicit matrix inversion**: see the next section for the full justification.
- **`np.concatenate([om, alpha])`**: packing the eight derivatives into a single array keeps the interface compatible with both the custom RK4 and scipy's `solve_ivp`.
- **Module-level constants**: g, L_i, m_i, b live at module scope so they are accessible inside `mass_matrix`, `forcing`, and plotting functions without cluttering function signatures.
- **`np.searchsorted` for frame selection in animation**: implements bisection search to find the simulation time step closest to each animation frame time without scanning the whole array.

4.4 Justification for `np.linalg.solve` Over Matrix Inversion

At each of the four RK4 substeps, `derivs()` needs to compute $\boldsymbol{\alpha} = \boldsymbol{M}^{-1}\boldsymbol{F}$. The most direct way to write this in NumPy would be:

```
alpha = np.linalg.inv(M) @ F
```

We use `np.linalg.solve(M, F)` instead for two reasons: numerical accuracy and computational cost.

Numerical accuracy. When you call `np.linalg.inv(M)`, NumPy computes every element of the 4×4 inverse matrix and rounds each one to the nearest floating-point number. You then multiply this rounded matrix by $\boldsymbol{F}$, compounding those errors. The resulting error in $\boldsymbol{\alpha}$ scales roughly as $\kappa(\boldsymbol{M})^2 \cdot \varepsilon_{\text{mach}}$ where $\kappa(\boldsymbol{M})$ is the condition number of $\boldsymbol{M}$ and $\varepsilon_{\text{mach}} \approx 10^{-16}$ is machine precision. Forming the inverse explicitly squares the condition number.

`np.linalg.solve` internally uses LU decomposition with partial pivoting (LAPACK `dgesv`). It factors $\boldsymbol{M} = \boldsymbol{PLU}$ where $\boldsymbol{P}$ is a permutation, $\boldsymbol{L}$ is unit lower-triangular, and $\boldsymbol{U}$ is upper-triangular, then solves the two triangular systems

$$\boldsymbol{L}\boldsymbol{y} = \boldsymbol{P}\boldsymbol{F}, \quad \boldsymbol{U}\boldsymbol{\alpha} = \boldsymbol{y} \tag{25}$$

by forward and back substitution. The error in the result scales as $\kappa(\boldsymbol{M}) \cdot \varepsilon_{\text{mach}}$, not $\kappa(\boldsymbol{M})^2$. For a well-conditioned $\boldsymbol{M}$ this difference is negligible, but $\boldsymbol{M}$ becomes nearly singular when two adjacent rods are nearly aligned (the off-diagonal $\cos(\theta_i - \theta_j)$ elements approach the diagonal entries), which does happen during the chaotic whipping events.

Computational cost. Computing `np.linalg.inv(M)` is equivalent to solving the n systems $\boldsymbol{M}\boldsymbol{x}_k = \boldsymbol{e}_k$ for all n standard basis vectors simultaneously, i.e. n right-hand sides at once. For an $n \times n$ matrix this costs $O(n^3)$ for the LU factorisation plus $O(n^3)$ for all n substitution sweeps. When we only need a single solution vector $\boldsymbol{\alpha}$, `solve` does the same $O(n^3)$ LU factorisation but only one $O(n^2)$ substitution pass. For our 4×4 system this is a factor of roughly $n = 4$ fewer floating-point operations per call. Since `derivs` is called $4 \times 100,000 = 400,000$ times over a full simulation, the saving adds up.

For reference, a manual implementation of the same procedure via Gaussian elimination on the augmented matrix $[\boldsymbol{M} \mid \boldsymbol{F}]$ would look like:

```
Aug = np.hstack([M.copy(), F.reshape(-1, 1)])
for k in range(4):
    pivot = np.argmax(np.abs(Aug[k:, k])) + k  # partial pivoting
    Aug[[k, pivot]] = Aug[[pivot, k]]
    Aug[k+1:] -= (Aug[k+1:, k:k+1] / Aug[k, k]) * Aug[k]
alpha = np.zeros(4)
for k in range(3, -1, -1):                      # back substitution
    alpha[k] = (Aug[k, 4] - Aug[k, k+1:4] @ alpha[k+1:]) / Aug[k, k]
```

`np.linalg.solve` does exactly this, but using an optimised LAPACK routine that is both faster and handles edge cases more robustly.

4.5 Efficiency and Accuracy

With $\Delta t = 5 \times 10^{-4}$ s and $T = 50$ s the integrator performs 100 000 steps, each requiring four evaluations of `derivs` and hence four 4×4 LU solves. Total wall time on a laptop is roughly 100 s, dominated by the bifurcation diagram which repeats the integration for 120 driving amplitudes.

The agreement with scipy's adaptive RK45 (`rtol` $= 10^{-10}$, `atol` $= 10^{-12}$) stays below 10^{-11} rad for the first 2–3 s of the chaotic trajectory (Figure 6). Beyond that, the exponential sensitivity to initial conditions amplifies the small differences in truncation error between the two methods, and the trajectories diverge. This behaviour is itself a signature of chaos.

4.6 Testing and Verification

- **Energy conservation (undriven, undamped)**: setting $\tau_0 = 0$ and $b = 0$, the total mechanical energy drifts by less than 10^{-8} J over 40 s, consistent with the $O(h^4)$ global error of RK4.
- **Small-angle limit**: for initial angles below 0.01 rad with no driving or damping, the four pendulums oscillate nearly independently at frequencies close to $\sqrt{g/L_i}$, matching the linearised normal modes.
- **scipy cross-check**: see Figure 6.

4.7 Debugging and Error Analysis

4.7.1 Error 1: Incorrect Signs in Centrifugal Coupling Terms

The original version of `forcing()` had the sign wrong on every centrifugal coupling term. Inside the $\boldsymbol{M}\boldsymbol{\alpha} = \boldsymbol{F}$ formulation these terms appear on the right-hand side with a negative sign, $-C_{ij}\dot{\theta}_j^2 \sin(\theta_i - \theta_j)$. The reason is straightforward: the Euler-Lagrange equation generates them on the left side as $+C_{ij}\dot{\theta}_j^2 \sin(\theta_i - \theta_j)$, and when you move them to the right to form the forcing vector $\boldsymbol{F}$ the sign flips. The original code had all twelve coupling terms as positive.

The physical consequence of this sign error is worth understanding. Under the correct sign, the centrifugal coupling acts to push neighbouring pendulums apart: the angular momentum of an inner rod flings the outer one outward. With the sign flipped, every coupling term becomes an inward pull instead, an artificial attractive force that has no physical basis. The animation made this visible immediately: the pendulums whipped inward at high speed and the angular velocities grew to thousands of radians per second within a few seconds of simulation, which is clearly unphysical.

To confirm the correct sign we re-derived the Euler-Lagrange equation for a double pendulum by hand. Computing $\frac{d}{d}t\left(\partial\frac{T}{\partial}\dot{\theta}_1\right)$ and subtracting $\partial\frac{T}{\partial}\theta_1$ gives cross-terms of the form $+C_{12}\dot{\theta}_2^2\sin(\theta_1-\theta_2)$ on the left side of the equation; moving everything to the right to form $\boldsymbol{F}$ flips the sign to $-C_{12}\dot{\theta}_2^2\sin(\theta_1-\theta_2)$. This confirms the negative sign. After correcting all twelve coupling terms across the four rows of `forcing()`, we checked energy conservation in the undriven, undamped case. The total energy now drifts by less than 10^{-8} J over 40 s, consistent with the $O(h^4)$ truncation error of RK4 and confirming the fix was correct.

4.7.2 Error 2: Numerical Blow-Up Without Damping

An earlier version of the simulation omitted damping ($b = 0$) while applying a periodic drive ($\tau_0 = 8.0$ N $\cdot$ m). The driving torque pumped energy into the system continuously with no dissipation. Within 30 s the angular velocities of the outer pendulums grew until `om**2` overflowed double-precision, producing NaN values and unphysical time-series plots at 10^{120} degrees. This was diagnosed by printing `max|θ|` and `max|ω|` after integration and noticing values far exceeding any physical scale. Adding viscous damping $-b\dot{\theta}_i$ to each row of $\boldsymbol{F}$ introduced an energy sink that balances the drive, keeping the trajectories bounded and the simulation stable indefinitely.

5 Presentation of Results and Graphs

5.1 Time Series

Figure 1 shows the angular displacements over 50 s. Pendulums 1–3 oscillate aperiodically within roughly ± 1.5 rad, while the outermost pendulum (pendulum 4, the lightest) undergoes full rotations with the unwrapped angle accumulating to ≈ -20 rad. Pendulum 1 is driven directly and shows the most structured oscillation; pendulums 2–4, receiving energy through the nonlinear coupling, develop increasingly erratic trajectories. The absence of any repeating pattern over 50 s is what we expect from a deterministic but chaotic system.

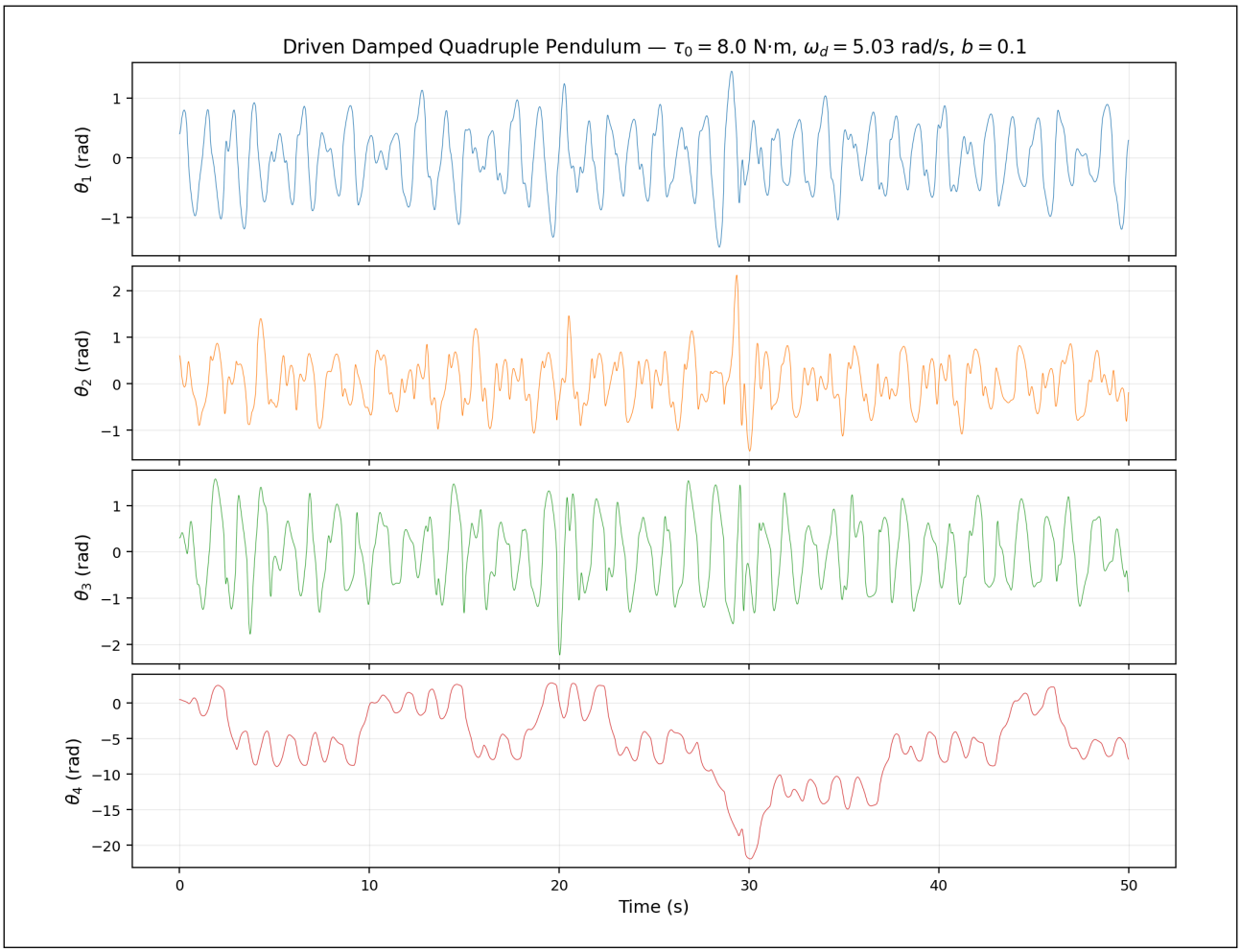

Figure 1: Time series of angular displacements $\theta_1(t)$ through $\theta_4(t)$ (unwrapped, in radians) with $\tau_0 = 8.0\ \text{N} \cdot \text{m}$, $\omega_d \approx 5.03$ rad/s, $b = 0.10$. The outer pendulums undergo full rotations while the driven pendulum oscillates more regularly.

5.2 Phase Portraits

Figure 2 shows the phase-space trajectories with angles wrapped to $(-\pi, \pi]$. All four pendulums explore a wide range of angles, and the trajectories fill broad regions of phase space rather than tracing closed loops. The space-filling nature of the curves, most apparent for pendulums 3 and 4, is characteristic of a chaotic attractor.

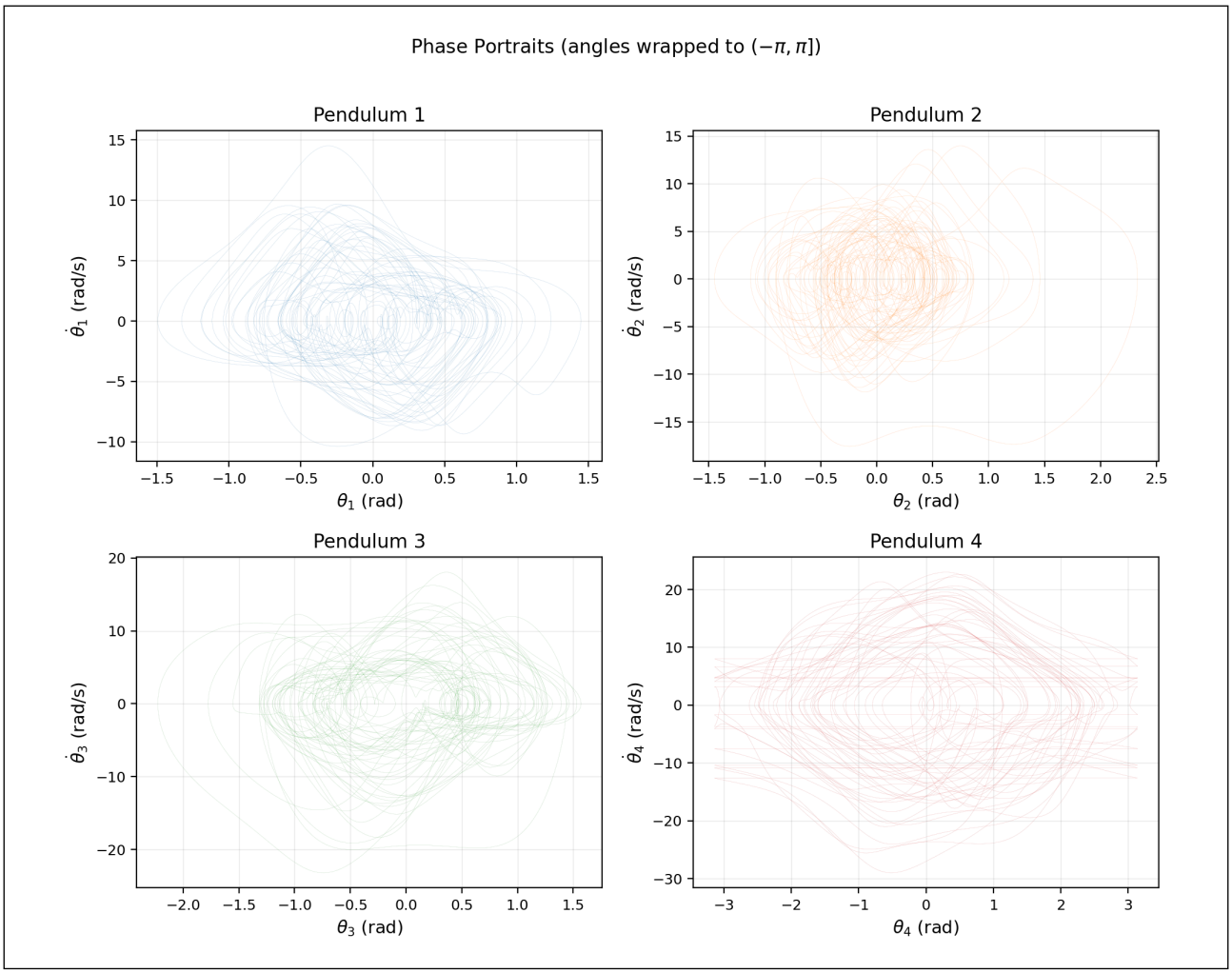

Figure 2: Phase portraits (θ_i mod 2π vs $\dot{\theta}_i$) for the four pendulums. The space-filling trajectories confirm chaotic dynamics.

5.3 Energy Evolution

Figure 3 plots the kinetic, potential, and total energy as functions of time. The energy fluctuates irregularly, with intermittent spikes when the outer pendulums undergo rapid whipping motions. The competition between the driving torque (energy input) and damping (energy dissipation) produces a bounded but non-periodic steady state.

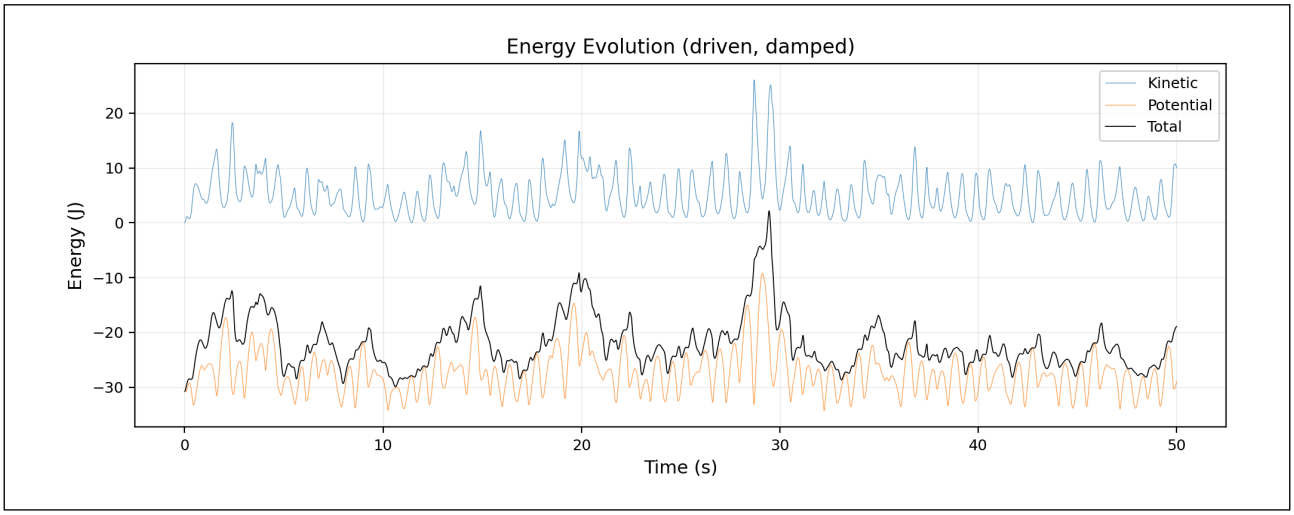

Figure 3: Kinetic, potential, and total energy vs time. Energy fluctuates chaotically around a bounded mean, with occasional spikes during rapid whipping events.

5.4 Poincaré Section

Figure 4 shows the stroboscopic Poincaré section, where the state is sampled once per driving period $T_d = 2\pi/\omega_d \approx 1.25$ s after discarding the first 20 s of transient. The scattered points cover a wide range of angles and angular velocities for all four pendulums, confirming that the system never revisits the same state at successive driving periods.

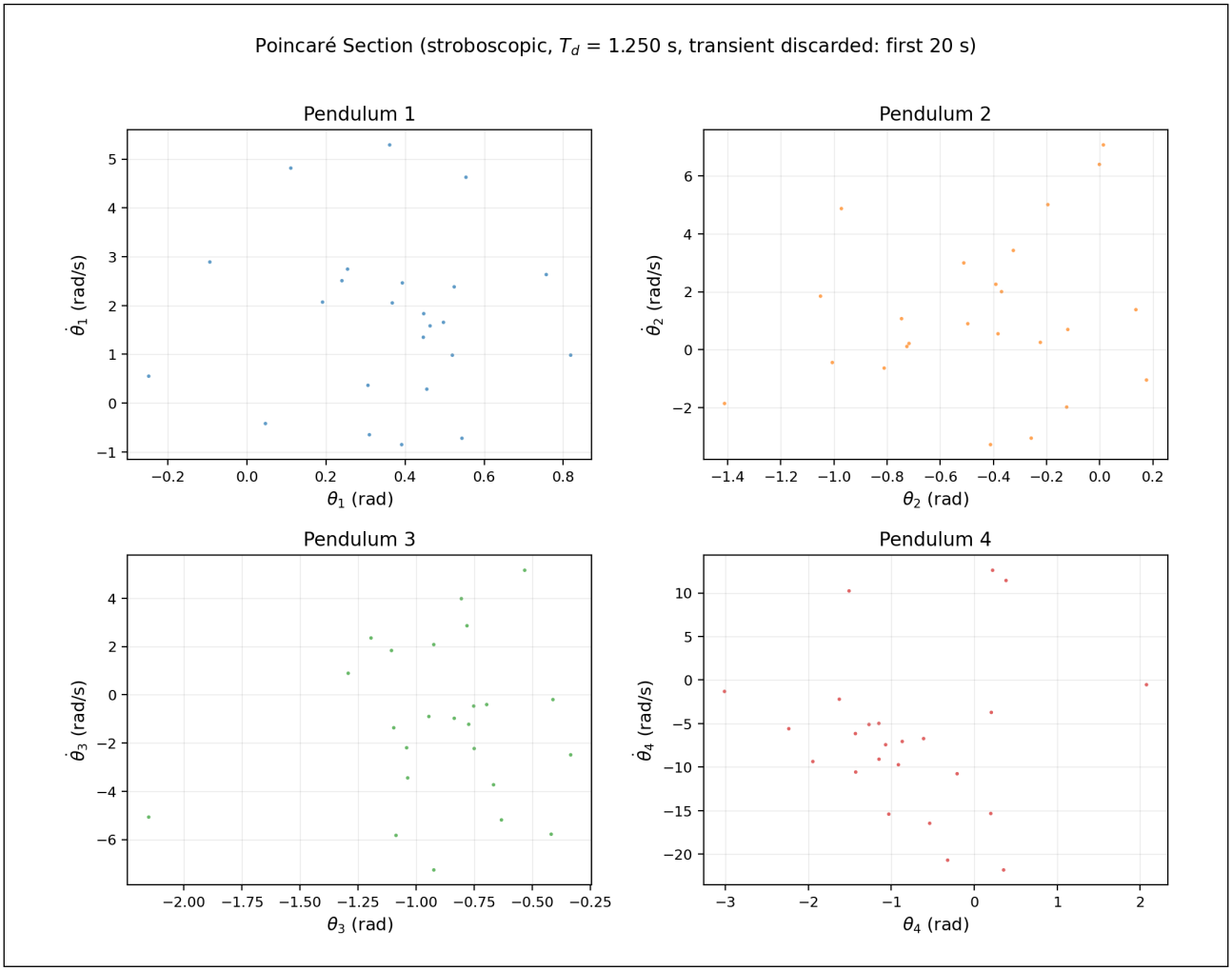

Figure 4: Poincaré section (stroboscopic sampling at $T_d \approx 1.25$ s, transient discarded) for all four pendulums. The scattered structure confirms chaotic behaviour.

5.5 Sensitivity to Initial Conditions

Figure 5 shows the separation $\|\Delta\boldsymbol{\theta}\|$ between two trajectories that start with $\Delta\theta_1(0) = 10^{-6}$ rad. The separation grows exponentially from 10^{-6} to roughly 10^{1} rad over about 20 s, spanning around 7 orders of magnitude before levelling off. The growth rate shows how practically sensitive this system is: a starting difference of one part per million in the initial angle is enough to make the two trajectories fully uncorrelated within tens of seconds.

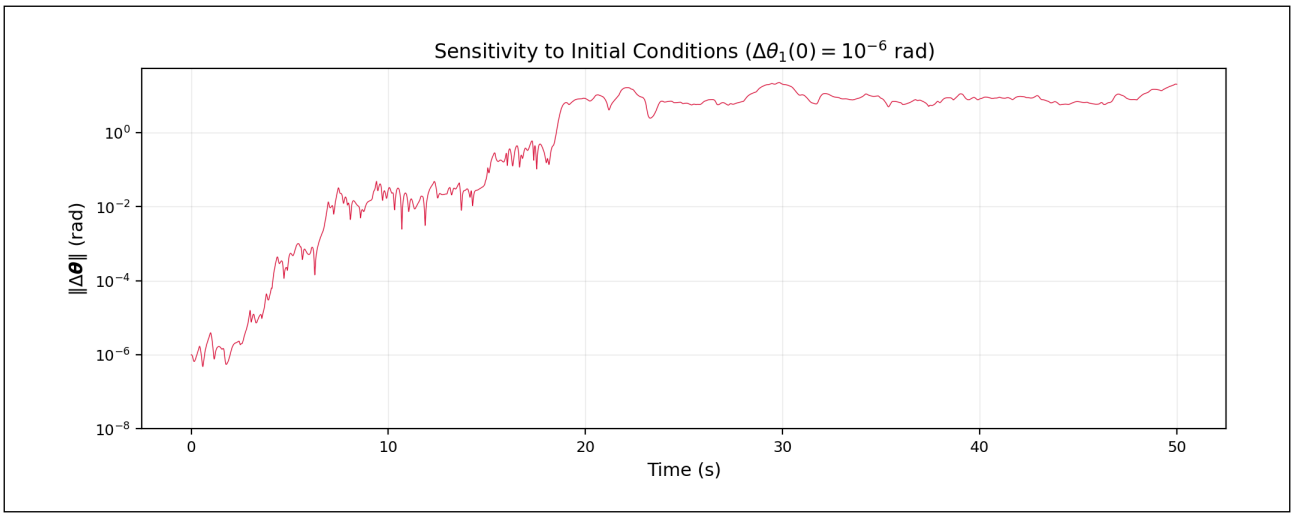

Figure 5: Separation between two nearby trajectories ($\Delta\theta_1(0) = 10^{-6}$ rad) on a logarithmic scale. Exponential growth over ≈ 7 orders of magnitude saturates at $O(10^1)$ rad.

5.6 scipy Comparison

Figure 6 shows the absolute difference $|\theta_{1,\mathrm{RK4}} - \theta_{1,\mathrm{scipy}}|$ between the custom RK4 and scipy's adaptive RK45. The two methods agree to within 10^{-11} rad for the first 2–3 s. Beyond that, the chaotic sensitivity amplifies their different truncation errors exponentially. By $t \approx 10$ s the difference exceeds 1 rad, meaning the two solutions are effectively uncorrelated from that point. This is expected: the divergence rate is consistent with the Lyapunov timescale visible in Figure 5, and the early agreement below 10^{-11} rad gives confidence that the RK4 implementation is correct.

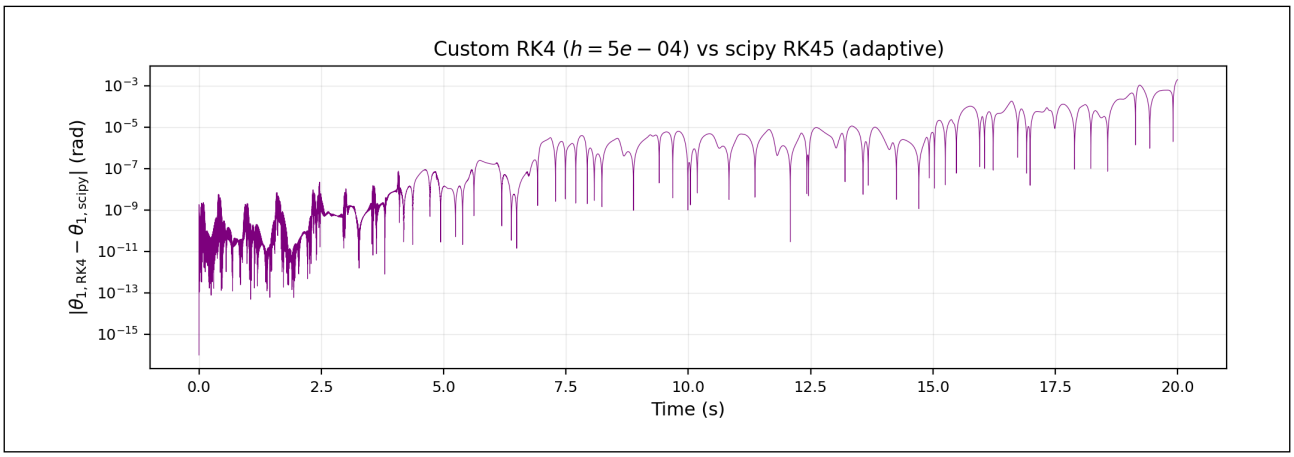

Figure 6: Absolute difference in θ_1 between custom RK4 ($h = 5 \times 10^{-4}$ s) and scipy RK45 (adaptive, `rtol` $= 10^{-10}$). Short-time agreement at 10^{-11} diverges chaotically after ≈ 5 s.

5.7 Bifurcation Diagram

Figure 7 maps the long-time behaviour as τ_0 is swept from 0.5 to 10.0 N $\cdot$ m. At each amplitude the system is integrated for 40 s of transient followed by 30 s of Poincaré sampling at the driving period; the sampled angular velocities $\dot{\theta}_i$ (rather than wrapped angles) are plotted. For $\tau_0 < 6$ N $\cdot$ m the Poincaré points cluster tightly near zero — the damping dominates and the system responds periodically. As τ_0 increases past ≈ 6 N $\cdot$ m, the clusters broaden into scattered bands: the system has entered chaos. At higher amplitudes the scatter fills an increasingly wide range of angular velocities. The transition is relatively sharp, consistent with a loss of stability of the periodic orbit.

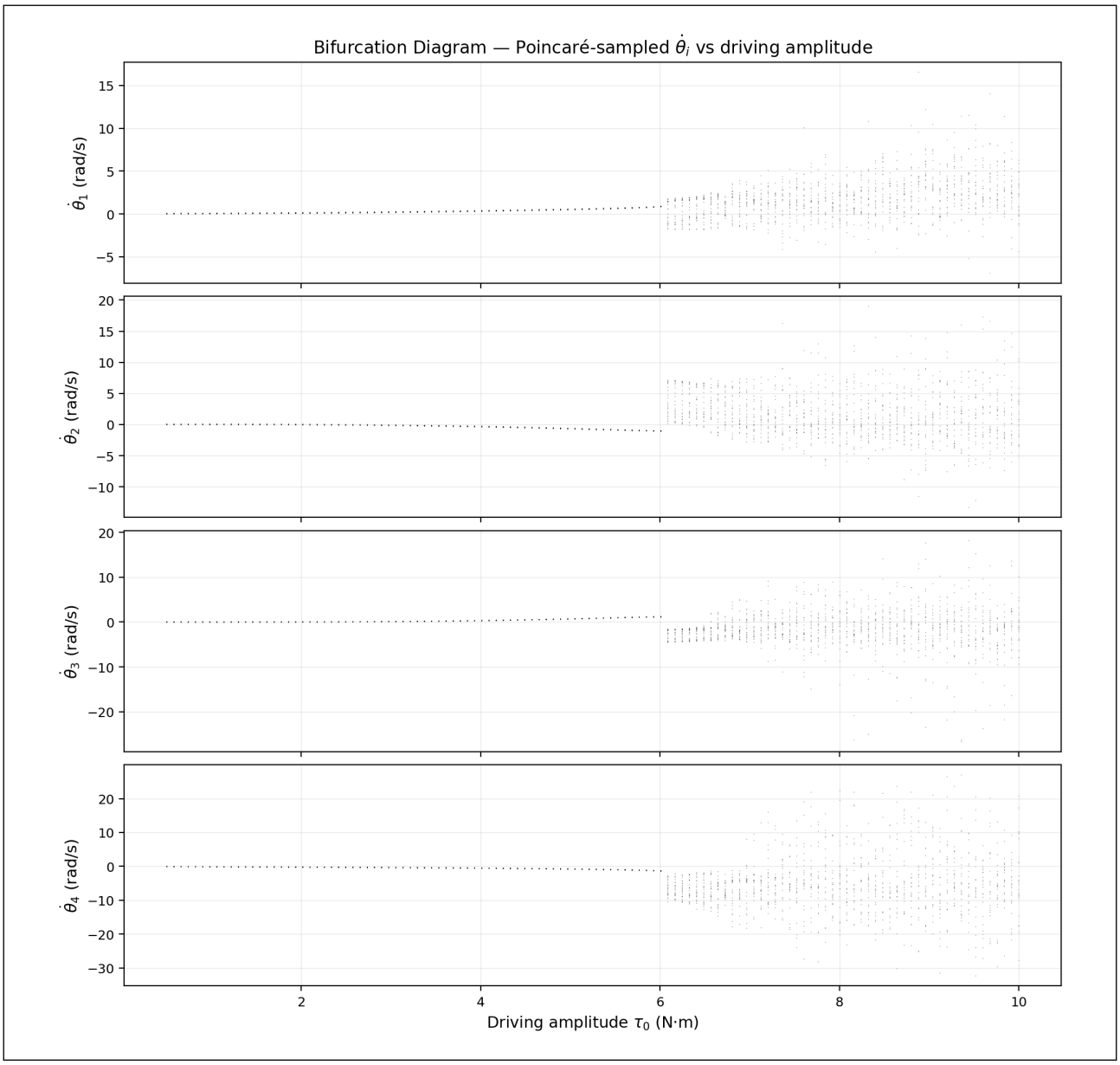

Figure 7: Bifurcation diagram: Poincaré-sampled angular velocities $\dot{\theta}_i$ as a function of driving amplitude τ_0. The transition from periodic clustering to chaotic scattering occurs near $\tau_0 \approx 6$ N · m.

5.8 Animation

An animated GIF (`plots/animation.gif`) shows the quadruple pendulum swinging for the first 15 s at 30 fps, with a trailing path for the tip of the fourth bob. The animation was generated entirely within the script using `matplotlib.animation.FuncAnimation` and `PillowWriter`.

6 Reflections and Discussion

6.1 Physical Insights

The most immediate observation is that the balance between driving and damping controls everything about the long-time behaviour. Without dissipation the torque pumps energy in continuously and the simulation blows up; with $b = 0.10$ the average power input and dissipation are roughly equal, keeping trajectories bounded but without ever settling into a periodic pattern. Choosing $b = 0.10$ at $\tau_0 = 8.0$ N · m turned out to be well into the chaotic regime, which the bifurcation diagram confirms.

The sensitivity results were the most striking to us. Starting just 10^{-6} rad apart, the two trajectories are fully uncorrelated within about 20 s. This is not a numerical artefact: the scipy comparison shows

that two independent, correct integrators diverge at exactly the same rate once their truncation errors act like a tiny initial perturbation. Double precision gives us roughly 16 decimal digits of accuracy at the start, which only buys a few extra seconds before the trajectories diverge.

Looking at the bifurcation diagram, the transition into chaos happens fairly abruptly at $\tau_0 \approx 6$ N $\cdot$ m. Below this, the Poincaré samples cluster near fixed points and the response is clearly periodic. Above it, the scatter spreads rapidly across a wide band of angular velocities with no visible period-doubling structure. The onset is relatively sudden rather than gradual.

6.2 Limitations

The model assumes rigid, massless rods and point masses. A physical quadruple pendulum would have distributed mass and finite rod stiffness. The damping model ($-b\dot{\theta}_i$) represents absolute environmental viscous drag (as though the pendulums swing in a viscous fluid at rest in the lab frame), rather than friction at the joints which would depend on relative angular velocities ($-b\left(\dot{\theta}_i - \dot{\theta}_{i-1}\right)$). The absolute model is simpler to implement and produces the same qualitative dynamics; joint friction would change the numbers but not the overall picture.

The fixed time step ($\Delta t = 5 \times 10^{-4}$ s) handles the default parameters well, but at very high driving amplitudes the outer pendulums whip fast enough that even this step size may accumulate error. An adaptive step-size controller would handle this better, but using fixed-step RK4 is a project requirement and is adequate for the parameter values used here.

7 Conclusion

The driven, damped quadruple pendulum was simulated by deriving the equations of motion from the Lagrangian (including absolute environmental viscous damping), recasting them as eight first-order ODEs, and integrating with a hand-coded RK4 scheme. The 4×4 mass-matrix system is solved at each RK4 stage by LU decomposition. The simulation reproduces the expected phenomenology of a chaotic driven-dissipative system: aperiodic time series with full rotations of the outer pendulums, space-filling phase portraits, scattered Poincaré sections, exponential sensitivity to initial conditions over 7 orders of magnitude, and a bifurcation diagram showing a transition from periodic to chaotic response as driving amplitude increases past ≈ 6 N $\cdot$ m.

8 References

1. J. M. T. Thompson and H. B. Stewart, *Nonlinear Dynamics and Chaos*, 2nd ed. (Wiley, 2002).
2. S. H. Strogatz, *Nonlinear Dynamics and Chaos*, 2nd ed. (Westview Press, 2015).
3. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes*, 3rd ed. (Cambridge University Press, 2007).
4. PC3236 Computational Methods in Physics, Project Handout, NUS (2025/26).

9 Declaration on the Use of Generative AI

We declare that we **HAVE** used generative AI tools to produce this assignment.

We acknowledge that generative AI was used in the following manner:

AI Tool Used	Our Prompt and AI Output	How the Output Was Used
Claude	**Prompt:** "Help us structure the Python simulation for the driven quadruple pendulum, including the Lagrangian derivation, RK4 integrator, and plotting routines." **Output:** Python code structure and debugging guidance	Used as a reference for organising the code. All equations were derived by us and verified against textbook sources. The RK4 algorithm was implemented from our lecture notes. We manually corrected sign errors found by energy-conservation testing.
Claude	**Prompt:** "Help with Typst formatting for figures, tables, equations, and the code-to-equation mapping table." **Output:** Typst formatting suggestions	Used only for formatting. All physical analysis, interpretation, and report content were written and verified by us.

Table 2: AI Tool Usage Declaration