

Problem Set #5 — Adaptive RK4 for Lorenz, Rossler Systems

University of Colorado - Boulder
CSCI4446 — Chaotic Dynamics

JACK MATHEWS
john.mathewsiii@colorado.edu

February 19, 2026

1 Adaptive RK4 Implementation

See Github Repository if interested in the specifics of the implementation.

My adaptive solver begins with a user-defined time-step dt . It then runs one step of RK4 with that time-step alongside two half steps, *i.e.* time-step = $\frac{dt}{2}$. The error between the two is evaluated by taking the 2-norm of their vector difference (error vector) and comparing that to the tolerance value $tol = 0.001$. If the error is greater than the tolerance, it sets dt to $\frac{dt}{2}$ and recursively repeats the process until it finds a time-step that does produce an error norm within the specified tolerance. However, if the error is greater than or equal to the tolerance, it checks if doubling the time step would be sufficiently accurate. It does this by evaluating two RK4 iterations using the original dt and comparing that result to that of one RK4 iteration of step size $dt \times 2$. If the error between those two is within the tolerance, it recursively repeats that step until it finds a time-step that produces error outside the tolerance and returns the result from the next smallest time step. Note also that I used the 2-norm instead of the ∞ -norm as it was more stable.

2 RK4 on the Lorenz System

2.1 Adaptive RK4 Results

Figure 1 shows the output of my adaptive RK4 method when applied to the Lorenz system with the following parameters:

1. $\{x_0, y_0, z_0\} = \{-13, -12, 52\}$
2. $\{a, b, r\} = \{16, 4, 45\}$
3. Iterations: 5000
4. Adaptation Tolerance: 0.001
5. Initial dt : 0.001

Lorenz State Space Plot

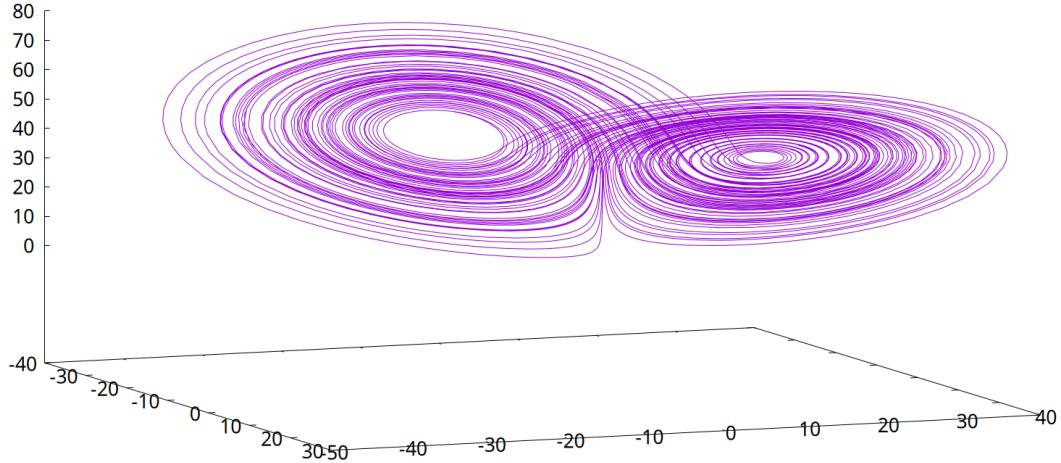


Figure 1: Lorenz System Simulated via Adaptive RK4

At first glance, this looks fairly accurate, so there is little to say here. Discussion of the accuracy is in the following subsection.

2.2 Comparison of Adaptive and Nonadaptive Methods

Figure 2 compares two Lorenz simulations, one adaptive (purple) and one fixed (green). As we can see from the gaps between the lines, the adaptive simulation often skips a lot of time. In fact, while these plots run to the same time $t_{end} = 1.871$ seconds, the adaptive method only ran 100 iterations, while the nonadaptive method ran for 1 871 iterations. However, despite the gaps in the plot, the two versions remained accurate to each other, as suggested by the fact that the purple line segments always begin and end on the green line. Finally, it is obvious that the adaptive solver's time steps are not evenly spaced. This is easier to see in 1, where the outside line segments are long enough to distinguish, while the inside line segments are often too small to distinguish, appearing to the human eye as a curve, rather than a set of discrete lines. This fact demonstrates that the adaptive solver is correctly identifying the spots that require more precision and shrinking the time step accordingly while doing the reverse for the spots that require less precision.

2.3 Parameter Exploration

The Lorenz system is highly dependent on the value of the parameter r . Keeping a and b the same while varying r , we can get interesting results. For example, when $0 \leq r \leq 1$, the system converges to the origin, seemingly independent of initial condition.

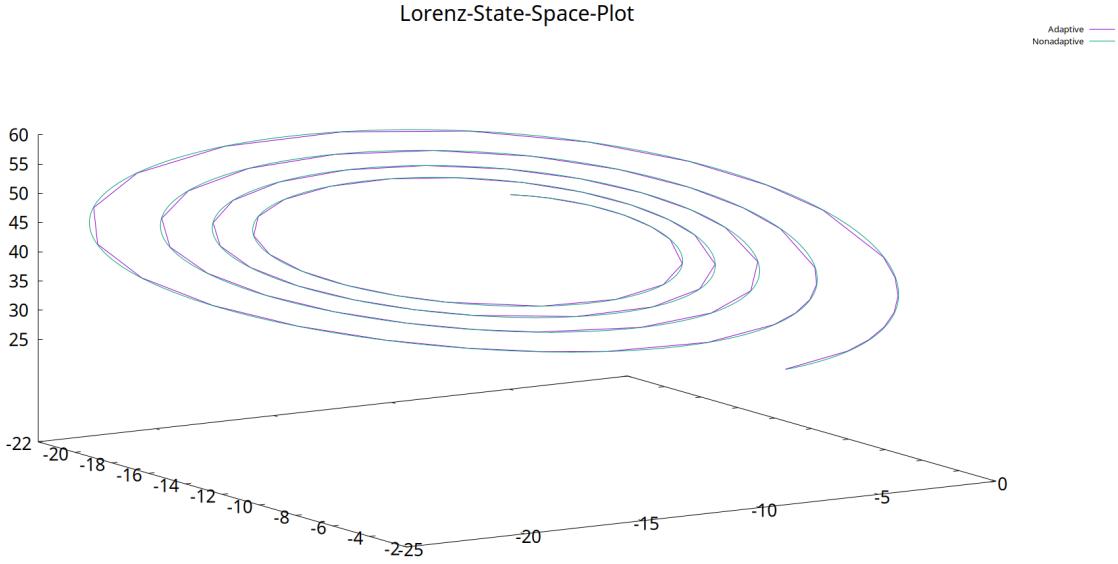


Figure 2: Adaptive vs. Nonadaptive Comparison

When $13.5 \leq r \leq 14$, the system converges to one of two fixed points, depending on the exact value of r . The first occurs at approximately $(-7.1045, -7.1043, 12.1768)$ and appears when $13.5 \leq r \leq 13.618$. The second occurs around $(7.1045, 7.1043, 12.1768)$ and appears when $13.618 \leq r \leq 14$. From this we can conclude there is a bifurcation from one fixed point to the other in the range $13.618 < r < 13.619$. Interestingly, the second fixed point is an exact reflection of the first across the z -plane. Furthermore, this bifurcation only appeared for initial conditions around those from the previous section. Different initial conditions produced different fixed points and/or bifurcations.

For the same initial conditions, there are further bifurcations between $(23 \leq r \leq 30)$. For $23 \leq r \leq 28.9171$, the system converged to a fixed point, although that fixed point was different for different r values. However, for $28.9172 \leq r$, the state-space plot more-or-less resembled the classic chaotic attractor from 1, indicating a bifurcation between $28.9171 < r < 28.9172$. For different initial conditions, the bifurcation appeared relatively close to this position, but its exact position in r -space is different.

Bifurcations based on r are theoretically independent of initial condition, but due to inaccuracies in the solver, the exact values of r where the bifurcations occur are obscured.

Rossler-State-Space-Plot

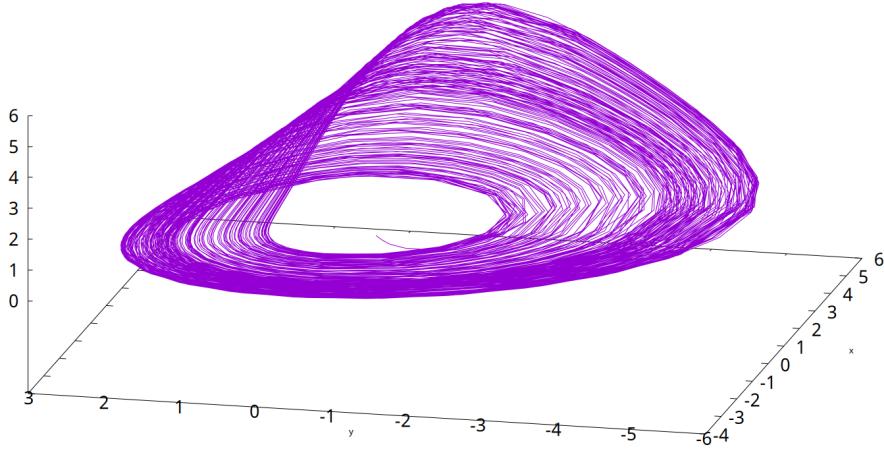


Figure 3: Rossler System State Space Plot

3 Rossler System

Figure 3 shows output from my adaptive solver on the Rossler system for the initial conditions and parameters:

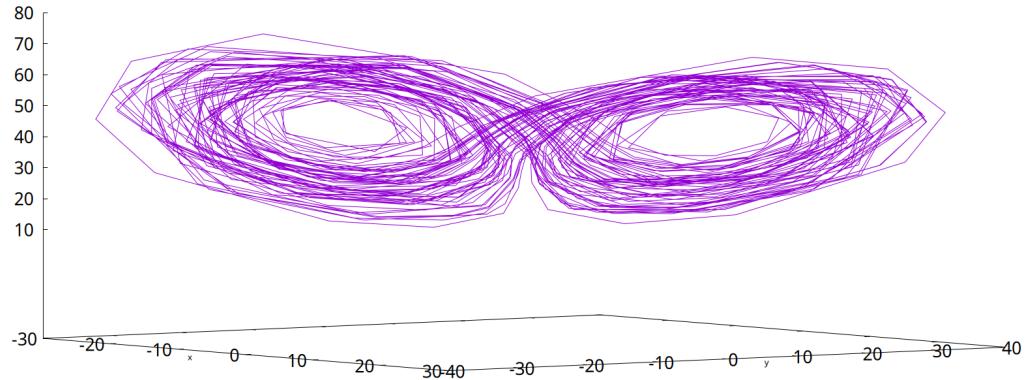
$$\{a, b, c\} = \{0.398, 2, 4\}$$

$$\{x_0, y_0, z_0\} = \{-1, -1, 1\}$$

4 Adaptation Tolerance Experiments

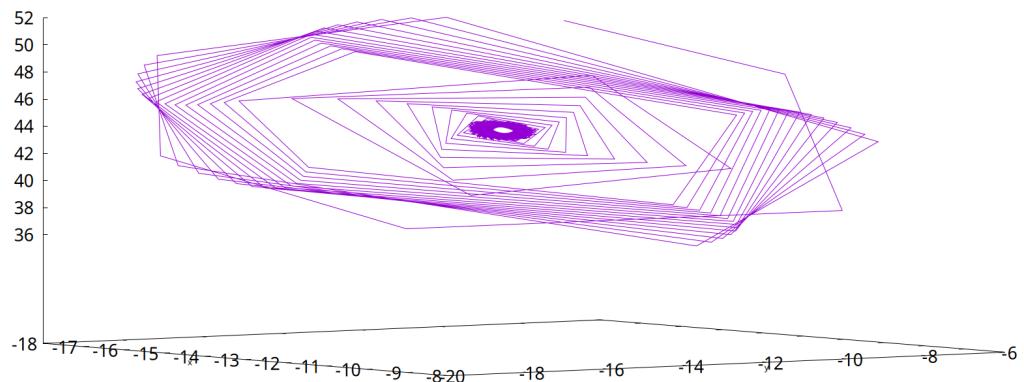
As shown before, setting the tolerance for the adaptation parameter to 0.001 is effective in maintaining accuracy, at least for short simulation durations. However, the solver can maintain the dynamics of the system for much higher tolerance values. Figure 4 demonstrates that the classic butterfly structure can remain intact even with $tol = 0.6$, though increasing it to just 0.7 results in a complete breakdown of the dynamics. Of course, setting the tolerance as high as six-tenths produces nearly useless data, but it is interesting that the solver can correctly produce the structure of the system for such an extended time—the timestamp at the final plotted point was 57.7 seconds.

Adaptive-Lorenz-State-Space-Plot



(a) $tol = 0.6$

Adaptive-Lorenz-State-Space-Plot



(b) $tol = 0.7$

Figure 4: Adaptive RK4 on the Lorenz System for Different Adaptation Tolerance Values