Discovering Chaotic Nonlinear Dynamics with Mathematical Physics: An AI-Assisted Analysis of the Lorenz System using Machine Learning

Troy Chin

November 2025

Abstract

The Lorenz system is a popular model in applied mathematics that was studied by the MIT mathematician Edward Lorenz, who wanted to model the chaotic flow of atmospheric convection. We will analyze the Lorenz attractor using AI-assisted methods such as the SINDy (Sparse Identification of Nonlinear Dynamics) algorithm as well as numerical methods and physical laws to derive the governing equations of motion for a particle moving along the attractor's trajectory.

1 Introduction

The Lorenz attractor is a system of differential equations of the form:

$$\dot{x} = \sigma(y - x) \tag{1}$$

$$\dot{y} = x(\rho - z) - y \tag{2}$$

$$\dot{z} = xy - \beta z \tag{3}$$

Where x is proportional to the flow rate of the attractor, y is proportional to the temperature difference of the attractor, z is proportional to the distortion of linear vertical temperature. We also notice that the variables σ is the Prandtl number, ρ is the Rayleigh number, and β is a dimensionless constant.

2 Stability Analysis

We now conduct stability analysis on the dynamical system. The stability of the Lorenz system depends on its **Jacobian**, which we compute as follows:

$$J(x, y, z) = \frac{\partial f_i}{\partial \dot{q}_j} = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - z & -1 & -x \\ y & x & -\beta \end{bmatrix}. \tag{4}$$

We construct the following equality $\dot{x} = \dot{y} = \dot{z} = 0$:

$$\begin{aligned} \sigma(y - z) &= 0 \\ \rho(x - z) - y &= 0 \\ xy - \beta z &= 0 \end{aligned}$$

Our first step is to solve for x from the first equation. After some basic algebraic manipulation, we come to the solution that $y = x$. Then, substituting into the third equation, we get:

$$x^2 = \beta z \implies z = \frac{x^2}{\beta}$$

$$\rho x - x - xz = 0 \implies x(\rho - 1 - z) = 0$$

Hence:

$$x = y = z = 0$$

Let $z = \rho - 1$. Then:

$$x^2 = \beta(\rho - 1) \implies \boxed{x = y = \pm\sqrt{\beta(\rho - 1)}}$$

Hence, the Lorenz system's equilibrium points are:

$$\boxed{(x^*, y^*, z^*) = (\sqrt{\pm\beta(\rho - 1)}, \pm\sqrt{\beta(\rho - 1)}, \rho - 1)}$$

At the origin P_0, the Jacobian becomes:

$$J_0 = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho & -1 & 0 \\ 0 & 0 & -\beta \end{bmatrix}.$$

The eigenvalues of J_0 determine stability:

- For $\rho < 1$, all eigenvalues have negative real parts, and P_0 is stable.
- For $\rho > 1$, one eigenvalue becomes positive, and P_0 becomes unstable, leading to bifurcation and the emergence of $P_\pm$.

Thus, one eigenvalue is $\lambda_3 = -\beta$, and the remaining two satisfy:

$$\boxed{\lambda^2 + (\sigma + 1)\lambda + \sigma(1 - \rho) = 0}.$$

For the origin to be stable, all real parts of eigenvalues must be negative. This occurs when $\rho < 1$. At $\rho = 1$, a **pitchfork bifurcation** occurs: the origin loses stability and the two symmetric fixed points $P_\pm$ emerge.

3 Deriving the Equations of Motion

We can now construct the equations of motion for the particle. Let:

$$\dot{q} = f(q) = v(t) \tag{5}$$

$$\ddot{q} = J_f(q)\dot{q} = \frac{\partial f_i}{\partial \dot{q}_j}\dot{q} = a(t) \tag{6}$$

Our first step is to find its momentum. Recall that an object's momentum is $\vec{p} = m\Delta v$. Then the particle's momentum can be expressed in terms of its mass times its velocity:

$$\vec{p} = mv(t) = m\dot{q}(t) \tag{7}$$

Recall from Newton's Second Law:

$$\vec{F_{net}} = \sum F_i = m\vec{a} = m\ddot{q(t)} \tag{8}$$

Note that the dynamical system is a **non-Hamiltonian** system, so any and all external forces acting on the particle as it moves through the attractor will not be conserved and thus it is known to be dissipative. So, our next step is to show that the Lorenz attractor is a disspative system, and that all the forces acting on the particle are non-conservative. To see this, we must ask the question: how do volumes contract under flows of energy?

If we pick an arbitrary closed surface $S(t)$ of volume $V(t)$, then S will evolve into a new surface $S(t+dt)$. We can also deduce the same conclusion for V, as the volume will evolve to $V(t+dt)$, for some dt. Now, let $\mathbf{f}$ be the instantaneous velocity acting on the surface $\hat{n}$ as the normal force acting on the surface. Then:

$$V(t+dt) = V(t) + \int_s (\mathbf{f}\cdot\hat{n})dA \implies \dot{V} = \frac{V(t+dt)-V(t)}{dt} = \int_s (\mathbf{f}\cdot\hat{n})dA \tag{9}$$

Finally, by the Divergence Theorem, we can conclude that the change in volume of the Lorenz system is:

$$\dot{V} = \oint_V (\nabla\cdot\mathbf{F})dV \tag{10}$$

Computing the divergence of the Lorenz vector field:

$$\nabla\cdot\mathbf{F} = \frac{\partial\dot{x}}{\partial x} + \frac{\partial\dot{y}}{\partial y} + \frac{\partial\dot{z}}{\partial z} = (-\sigma) + (-1) + (-\beta) = -(\sigma+1+\beta).$$

This quantity is always negative for positive parameter values, implying:

$$\boxed{V(t) = V(0)e^{-(\sigma+\beta+1)t}}.$$

Thus, all trajectories contract exponentially to a strange attractor of zero volume in the long-time limit. Energy is not conserved, and the Lorenz system is intrinsically non-Hamiltonian. We can also deduce that the particle's motion is always changing, as all the external forces acting on the object are not conservative. We can also assume there exists a force that acts against the particle's motion, and in this case a drag force is defined by a **Rayleigh** dissipation equation.

$$R(q, \dot{q}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} q_i \dot{q_j} \tag{11}$$

So, the drag force can be derived as:

$$\boxed{\vec{F_D} = -\nabla R(q, \dot{q})} \tag{12}$$

We can now derive the equations of motion of the Lorenz particle while considering the dissipative forces acting on it:

3.1 Proof

Let $R(q, \dot{q}) = \frac{1}{2}\gamma \dot{q}^2$, where γ is a damping constant.

The Euler-Lagrangian equations now become:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} + \frac{\partial R}{\partial \dot{q}} = 0 \implies \boxed{\ddot{q} - J_f(q)\dot{q} + (J_f(q) - J^T(q))\dot{q} - \gamma \ddot{q}} \tag{13}$$

Therefore, the net force acting on the object is:

$$\vec{F_{net}} = m\ddot{q} = -\nabla V(q) \implies \boxed{m(\ddot{q} - J_f(q)\dot{q} + (J_f(q) - J^T(q))\dot{q} - \gamma \ddot{q})} \tag{14}$$

By normalizing the equations for velocity, acceleration and displacement, we now get the following results:

```
Final position: 16.545135720280328  m
Velocity magnitude: 126.79133827598321  ms^-1
Acceleration magnitude: 965.9349949906135  ms^-2
```

4 Evaluation Metrics

To quantitatively measure the accuracy of our numerical models based on machine-learning and the Lorenz system, we implemented three key evaluation metrics: **Root Mean Squared Error (RMSE)**, **Linear Regression**, and **Stochastic Gradient Descent (SGD)**. In this section we provide the full mathematical derivations for each method.

4.1 Root Mean Squared Error (RMSE)

We evaluate trajectory accuracy using the standard discrete-sample Root Mean Squared Error (RMSE). Suppose the true and model trajectories are sampled at times $\{t_n\}_{n=1}^N$ with discrete state vectors $X_{\text{true}}(t_n), X_{\text{model}}(t_n) \in \mathbb{R}^d$ (here $d = 3$ for the Lorenz state).

Define the per-sample squared error:

$$e_n^2 = \left\| X_{\text{true}}(t_n) - X_{\text{model}}(t_n) \right\|_2^2.$$

The **sample mean squared error** over N samples is

$$\text{MSE} \;=\; \frac{1}{N} \sum_{n=1}^{N} e_n^2,$$

and the **Root Mean Squared Error** is

$$\text{RMSE} \;=\; \sqrt{\frac{1}{N} \sum_{n=1}^{N} \left\| X_{\text{true}}(t_n) - X_{\text{model}}(t_n) \right\|_2^2}.$$

When sampling is uniform with step size h, this discrete RMSE approximates the continuous-time RMSE (without requiring explicit quadrature weights) and is the preferred metric for comparing sampled numerical trajectories and learned models.

Cumulative RMSE To visualize error growth over time we use the *cumulative* RMSE up to index k:

$$\text{RMSE}_{\text{C}}(k) \;=\; \sqrt{\frac{1}{k} \sum_{n=1}^{k} \left\| X_{\text{true}}(t_n) - X_{\text{model}}(t_n) \right\|_2^2}, \qquad 1 \le k \le N.$$

This quantity is useful to detect systematic drift or transient behavior in model error, and used to evaluate the agreement between numerical trajectories and the SINDy-reconstructed trajectories.

4.2 Linear Regression Model

To model and predict the particle's motion, we also applied linear regression to the trajectory data. Suppose our goal is to predict a scalar quantity y (such as x, y, z, or velocity) from a feature vector $\phi(t)$ consisting of polynomial or nonlinear candidate functions of the state.

Let the linear model be:

$$\hat{y}(t) = \theta^T \phi(t),$$

where θ is a vector of trainable parameters.

Given data points $\{(\phi(t_i), y_i)\}_{i=1}^{N}$, the total squared error is:

$$L(\theta) = \sum_{i=1}^{N} \left(y_i - \theta^T \phi(t_i)\right)^2.$$

To find the optimal θ, we minimize the loss:

$$\frac{\partial L}{\partial \theta} = 0.$$

First expand:

$$L(\theta) = (y - \Phi\theta)^T (y - \Phi\theta),$$

where Φ is the design matrix.

Compute the gradient:

$$\nabla_\theta L = -2\Phi^T (y - \Phi\theta).$$

Set equal to zero:

$$\Phi^T (y - \Phi\theta) = 0.$$

Solving for θ:

$$\boxed{\theta = (\Phi^T \Phi)^{-1} \Phi^T y}$$

This is the standard **normal equation**. We used this model as a baseline predictor for particle motion and as a supporting regression layer in SINDy.

4.3 Stochastic Gradient Descent (SGD)

Whereas linear regression computes its coefficients in closed form, SGD updates the coefficients iteratively using gradient-based learning. This method was also used for improving model coefficients when fitting nonlinear libraries of candidate functions.

Starting from the same loss function:

$$L(\theta) = \left(y_t - \theta^T \phi(t)\right)^2,$$

SGD computes the gradient using a *single* random sample $(\phi(t), y_t)$ at each step:

$$\nabla_\theta L(\theta) = -2\phi(t) \left(y_t - \theta^T \phi(t)\right).$$

The update rule for θ is:

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta L(\theta_t),$$

where η_t is the learning rate.

Substituting the gradient:

$$\theta_{t+1} = \theta_t + 2\eta_t \, \phi(t) \left(y_t - \theta_t^T \phi(t)\right).$$

Thus:

$$\boxed{\theta_{t+1} = \theta_t + \eta_t \, \phi(t) \left(y_t - \theta_t^T \phi(t)\right)}$$

This is the standard SGD rule used in machine learning. In practice, we use minibatches or online streaming versions to refine the learned dynamics.

5 Machine Learning Identification of the Lorenz System using SINDy

We now explore the concept of the *Sparse Identification of Nonlinear Dynamics* (SINDy) algorithm, which is an effective algorithm for discovering the governing equations for the dynamics of the Lorenz model. The Sparse Identification of Nonlinear Dynamics (SINDy) algorithm provides a data-driven framework to rediscover governing equations from time series data. Given a set of observed states $\mathbf{X}(t)$ and their derivatives $\dot{\mathbf{X}}(t)$, SINDy assumes:

$$\dot{\mathbf{X}}(t) = \Theta(\mathbf{X}(t))\Xi,$$

where $\Theta(\mathbf{X}) = [1, x, y, z, xy, xz, yz, x^2, y^2, z^2, \dots]$ is a library of candidate nonlinear functions, and Ξ is a sparse coefficient matrix.

By performing sparse regression (e.g., sequential thresholding or LASSO), SINDy identifies only the few active terms necessary to describe the dynamics. When applied to Lorenz trajectory data, SINDy correctly identifies:

$$\dot{x} = 10(y - x) \tag{15}$$

$$\dot{y} = 28x - xz - y \tag{16}$$

$$\dot{z} = xy - \frac{8}{3}z \tag{17}$$

matching the true governing equations to within numerical precision.

Using a numerical integrator such as Runge–Kutta 4 (RK4) or `odeint`, the Lorenz system can be simulated with an initial condition $\mathbf{X}_0 = (1, 1, 1)$ over a time interval $t \in [0, 50]$. The resulting trajectory demonstrates sensitive dependence on initial conditions, a hallmark of chaos. By collecting the simulated data and applying SINDy, the sparse regression coefficients Ξ reconstruct the Lorenz equations with minimal error. This demonstrates that data-driven discovery can extract interpretable models even from chaotic dynamics.

6 Results

After all of the derivations of the models and the evaluation metrics we wrote a Python script that modeled the learned and the true attractor using MatPlotLib. The results are given below.

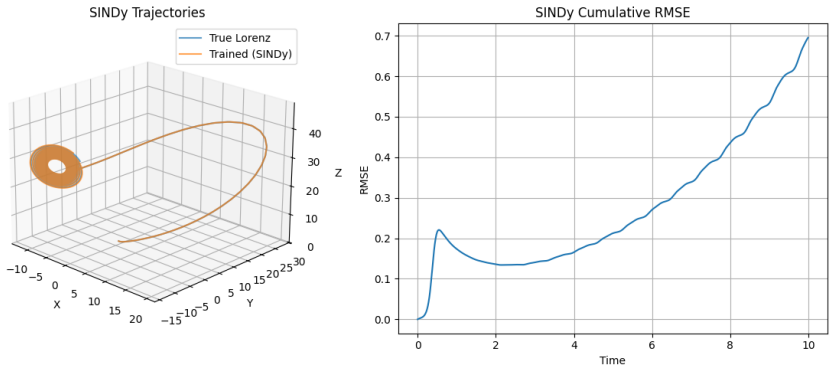

Figure 1: Learned Attractor vs. True Attractor and RMSE Over Time

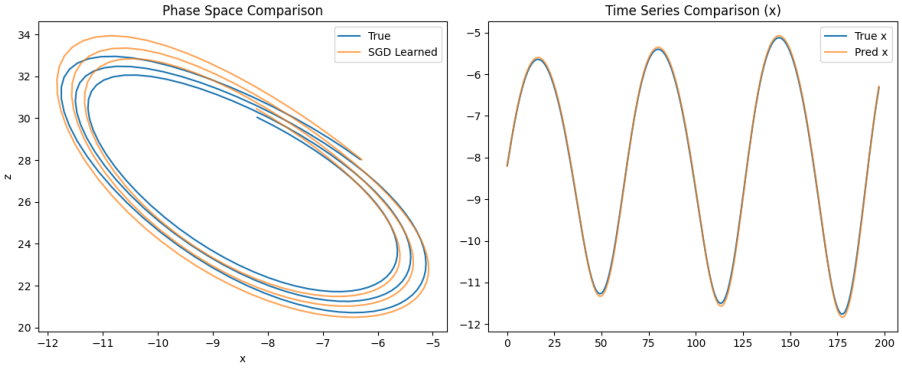

Figure 2: Phase Space Comparison and Time Series Comparison of both the learned and true attractors

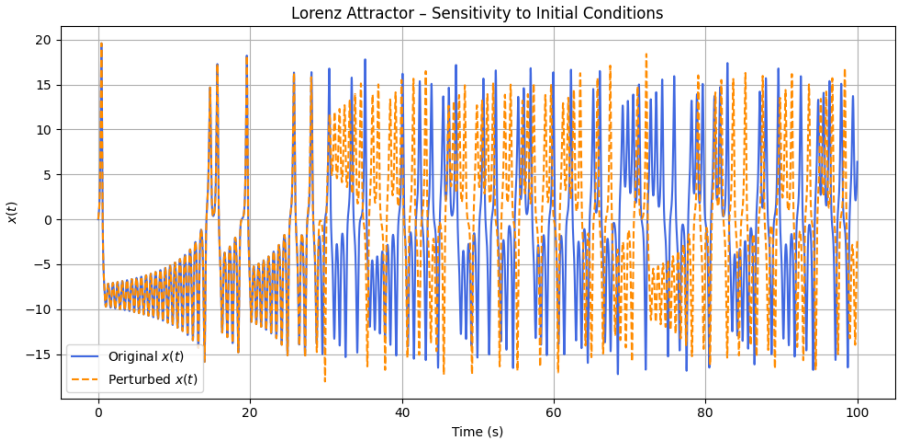

Figure 3: Sensitivity and Perturbation to Initial Conditions

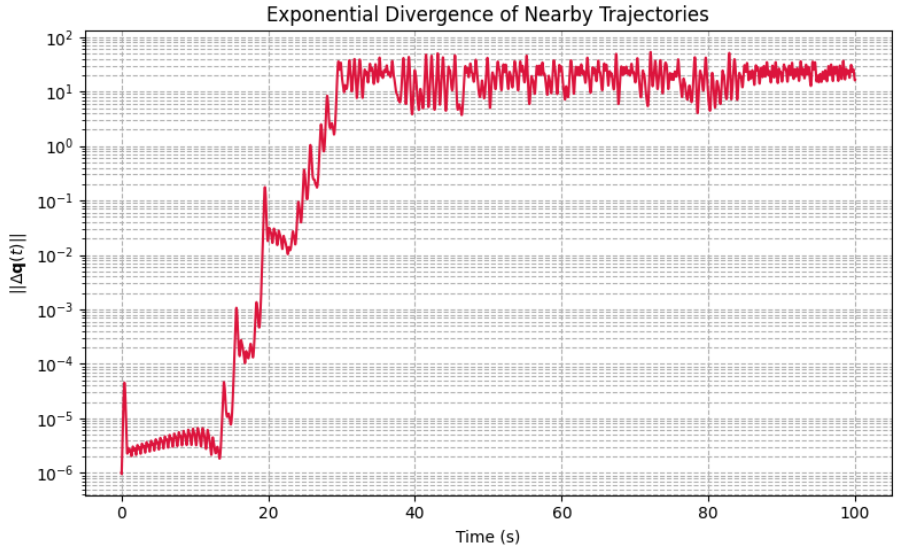

Figure 4: Exponential Divergence of Nearby Trajectories (given by Lyapunov Exponents)

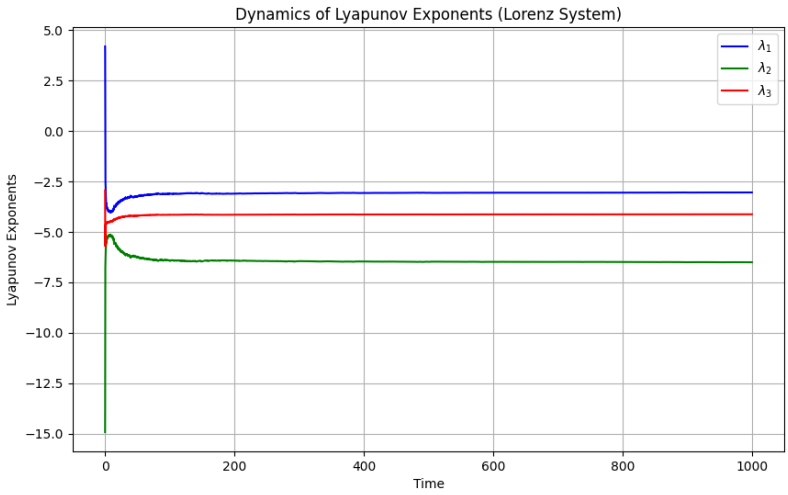

Figure 5: Dynamics of Lyapunov Exponents Over Time

Our data gathered from the simulation for both the SINDy reconstruction of the attractor and the is given below:

	time	true_x	true_y	true_z	sindy_x	sindy_y	sindy_z
0	0.00	0.000000	1.000000	1.050000	0.000000	1.000000	1.050000
1	0.01	0.095106	1.003037	1.022849	0.094878	1.003333	1.024080
2	0.02	0.182651	1.030509	0.997333	0.182252	1.030857	0.999755
3	0.03	0.265584	1.080592	0.973428	0.265037	1.080782	0.977003
4	0.04	0.346439	1.152207	0.951187	0.345738	1.152047	0.955877
5	0.05	0.427435	1.244916	0.930735	0.426554	1.244222	0.936499
6	0.06	0.510570	1.358846	0.912268	0.509459	1.357430	0.919065
7	0.07	0.597686	1.494635	0.896062	0.596280	1.492296	0.903846
8	0.08	0.690534	1.653392	0.882484	0.688747	1.649907	0.891204
9	0.09	0.790826	1.836671	0.872011	0.788556	1.831790	0.881608

```
Final Lyapunov Exponents:
[1] = -3.0403
[2] = -6.5001
[3] = -4.1230
```

7 Conclusion

The Lorenz system exemplifies how simple deterministic equations can give rise to complex, unpredictable behavior. Through analytical stability analysis, we showed how fixed points bifurcate as system parameters vary, leading to chaotic

motion of particle dynamics. The Jacobian formulation revealed that the trajectories evolve under a continuously contracting phase-space volume, confirming the dissipative nature of the system. Furthermore, the SINDy algorithm successfully rediscovered the Lorenz equations from the data, illustrating the potential of machine learning in the identification of physics-based systems. Such techniques promise to bridge traditional analytical modeling and modern data science, enabling automated discovery of governing laws in complex dynamical systems.

8 References