Project Plan

Chaos Theory Laboratory

Computational Exploration of Dynamical Systems

Project Code:	CTL-2024-001
Version:	1.0
Duration:	8–10 Weeks
Technologies:	Python, NumPy, Pandas, Matplotlib

Date: December 28, 2025

Document Control

lightgray **Title**	Chaos Theory Laboratory – Project Plan
Reference	CTL-2024-001
Version	1.0
Status	Draft
Author	Dikshant Neupane

Revision History

lightgray **Version**	**Date**	**Description**
1.0	December 28, 2025	Initial release

Contents

Chapter 1

Executive Summary

1.1 Purpose

This document presents the project plan for developing the **Chaos Theory Laboratory**—a computational platform for simulating, analyzing, and visualizing chaotic dynamical systems.

1.2 Objectives

OBJ-1 Develop numerical solvers (Euler, RK4) without external libraries

OBJ-2 Implement minimum three chaotic systems (Lorenz, Rössler, Logistic Map)

OBJ-3 Create analysis tools for Lyapunov exponents and bifurcation diagrams

OBJ-4 Generate publication-quality visualizations

OBJ-5 Demonstrate the Butterfly Effect quantitatively

1.3 Key Metrics

lightgray **Metric**	**Description**	**Target**
Duration	Total project timeline	8–10 weeks
Phases	Development phases	9
Milestones	Key checkpoints	4
Visualizations	Publication-quality figures	6
Systems	Chaotic systems to implement	3–5

1.4 Scope

In Scope: Custom numerical solvers, 3–5 chaotic systems, Lyapunov analysis, bifurcation diagrams, 2D/3D visualizations, documentation.

Out of Scope: External ODE solvers (scipy.odeint), PDEs, real-time GUI, web deployment.

Chapter 2

Technical Background

2.1 Chaos Theory

Chaos describes deterministic systems exhibiting sensitive dependence on initial conditions. Properties:

- **Deterministic:** Future states fully determined by initial conditions
- **Sensitive:** Small perturbations grow exponentially
- **Bounded:** Trajectories remain in finite region
- **Aperiodic:** Behavior never exactly repeats

2.2 Key Equations

2.2.1 Lorenz System

$$\dot{x} = \sigma(y - x) \qquad \dot{y} = x(\rho - z) - y \qquad \dot{z} = xy - \beta z \tag{2.1}$$

Parameters: $\sigma = 10$, $\rho = 28$, $\beta = 8/3$

2.2.2 Logistic Map

$$x_{n+1} = r \cdot x_n \cdot (1 - x_n) \tag{2.2}$$

2.2.3 Lyapunov Exponent

$$\lambda = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} \ln |f'(x_n)| \tag{2.3}$$

Interpretation: $\lambda > 0$ indicates chaos.

2.2.4 Runge-Kutta 4th Order

$$k_1 = f(x_n) \qquad k_2 = f(x_n + \tfrac{h}{2}k_1) \tag{2.4}$$

$$k_3 = f(x_n + \tfrac{h}{2}k_2) \qquad k_4 = f(x_n + hk_3) \tag{2.5}$$

$$x_{n+1} = x_n + \tfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{2.6}$$

Chapter 3

Project Timeline

3.1 Phase Overview

lightgray **Phase**	**Title**	**Duration**	**Week**	
0	Prerequisites Verification	2–3 days	Pre	
1	Mathematical Foundation	5 days	1	
2	Numerical Solver Development	7 days	2	
3	Multi-Variable Systems (Lorenz)	7 days	3	
4	Sensitivity Analysis (Butterfly Effect)	7 days	4	
5	Bifurcation Analysis	7 days	5	
6	Lyapunov Exponent Computation	7 days	6	
7	Extended Systems	7 days	7	
8	Integration & Documentation	7 days	8	

3.2 Milestones

lightgray **ID**	**Name**	**Week**	**Criteria**
MS-1	Solvers Complete	2	Euler and RK4 validated
MS-2	Core Systems	4	Lorenz + Butterfly Effect done
MS-3	Analysis Tools	6	Bifurcation + Lyapunov done
MS-4	Project Complete	8	All deliverables finalized

Chapter 4

Phase Details

4.1 Phase 0–1: Foundation

Phase 0: Prerequisites (2–3 days)

- ☐ Verify NumPy, Pandas, Matplotlib proficiency
- ☐ Configure development environment

Phase 1: Mathematical Foundation (Week 1)

- ☐ Study differential equations conceptually
- ☐ Understand numerical method theory
- ☐ Write pseudocode for solution approach

4.2 Phase 2: Numerical Solvers

Duration: Week 2 **Milestone:** MS-1

lightgray **Task**	**Description**	**Status**
T-2.1	Implement Euler method	☐
T-2.2	Implement RK4 method	☐
T-2.3	Validate against $dx/dt = -x$	☐
T-2.4	Generate comparison visualization	☐

Deliverables: `euler.py`, `runge_kutta.py`, `solver_comparison.png`

4.3 Phase 3: Lorenz System

Duration: Week 3

lightgray **Task**	**Description**	**Status**
T-3.1	Extend solvers for vector states	☐
T-3.2	Implement Lorenz equations	☐
T-3.3	Generate 3D attractor visualization	☐

Deliverables: `lorenz.py`, `lorenz_attractor.png`

4.4 Phase 4: Butterfly Effect

Duration: Week 4 **Milestone:** MS-2

lightgray **Task**	**Description**	**Status**
T-4.1	Run dual simulations with $\delta = 10^{-6}$	☐
T-4.2	Calculate trajectory divergence	☐
T-4.3	Create multi-panel visualization	☐
T-4.4	Estimate Lyapunov exponent from divergence	☐

Deliverables: `sensitivity.py`, `butterfly_effect.png`

4.5 Phase 5: Bifurcation Analysis

Duration: Week 5

lightgray **Task**	**Description**	**Status**
T-5.1	Implement logistic map	☐
T-5.2	Generate bifurcation diagram ($r \in [2.5, 4]$)	☐
T-5.3	Analyze period-doubling route to chaos	☐

Deliverables: `logistic.py`, `bifurcation.py`, `bifurcation_diagram.png`

4.6 Phase 6: Lyapunov Exponents

Duration: Week 6 **Milestone:** MS-3

lightgray **Task**	**Description**	**Status**
T-6.1	Implement Lyapunov calculator for logistic map	☐
T-6.2	Generate λ vs r diagram	☐
T-6.3	Validate correlation with bifurcation	☐

Deliverables: `lyapunov.py`, `lyapunov_vs_r.png`

4.7 Phase 7–8: Integration

Phase 7: Extended Systems (Week 7)

- ☐ Implement Rössler system
- ☐ Create attractor gallery visualization

Phase 8: Documentation (Week 8) **Milestone:** MS-4

- ☐ Organize code into modular structure
- ☐ Add docstrings to all functions
- ☐ Create README with usage instructions
- ☐ Generate final visualizations

Chapter 5

Deliverables

5.1 Code Modules

lightgray **Directory**	**File**	**Description**
solvers/	euler.py	Euler method
solvers/	runge_kutta.py	RK4 method
systems/	lorenz.py	Lorenz system
systems/	rossler.py	Rössler system
systems/	logistic.py	Logistic map
analysis/	lyapunov.py	Lyapunov calculator
analysis/	bifurcation.py	Bifurcation generator
analysis/	sensitivity.py	Butterfly effect

5.2 Visualizations

lightgray **Figure**	**Filename**	**Phase**	
Solver Comparison	solver_comparison.png	2	
Lorenz Attractor	lorenz_attractor.png	3	
Butterfly Effect	butterfly_effect.png	4	
Bifurcation Diagram	bifurcation_diagram.png	5	
Lyapunov Diagram	lyapunov_vs_r.png	6	
Attractor Gallery	attractor_gallery.png	7	

5.3 Project Structure

```
chaos_laboratory/
 solvers/
 systems/
 analysis/
 visualization/
 figures/
 data/
 main.py
 requirements.txt
 README.md
```

Chapter 6

Risk Management

lightgray **Risk**	**Prob.**	**Impact**	**Mitigation**
Mathematical complexity	Med	High	Extra study time in Phase 1; use recommended resources
Numerical accuracy issues	Med	High	Validate against analytical solutions; compare methods
Visualization quality	Low	Med	Study documentation; iterate on parameters
Scope creep	Med	Med	Defer extensions to optional Phase 9

Chapter 7

Quality & Resources

7.1 Quality Standards

Code: All functions documented with docstrings; PEP 8 naming conventions.
Visualizations: 300 DPI; labeled axes; descriptive titles.
Validation: Solvers tested against $dx/dt = -x$ (exact: e^{-t}); error $< 1\%$.

7.2 Success Criteria

- ☐ Numerical solvers produce $< 1\%$ error on test cases
- ☐ Minimum 3 chaotic systems fully functional
- ☐ All 6 required visualizations generated at 300 DPI
- ☐ All functions include docstrings
- ☐ README complete with installation and usage
- ☐ Butterfly effect quantitatively demonstrated

7.3 Resources

Software: Python 3.8+, NumPy, Pandas, Matplotlib, Git
References:

- Gleick, J. *Chaos: Making a New Science*
- Strogatz, S.H. *Nonlinear Dynamics and Chaos*
- Wikipedia: Lorenz system, Logistic map, Lyapunov exponent

Appendix A

Reference Equations

Lorenz: $\dot{x} = \sigma(y - x)$, $\dot{y} = x(\rho - z) - y$, $\dot{z} = xy - \beta z \quad (\sigma = 10, \rho = 28, \beta = 8/3)$
Rössler: $\dot{x} = -y - z$, $\dot{y} = x + ay$, $\dot{z} = b + z(x - c) \quad (a = 0.2, b = 0.2, c = 5.7)$
Logistic: $x_{n+1} = rx_n(1 - x_n) \quad (r \in [2.5, 4])$
Lyapunov: $\lambda = \frac{1}{N}\sum_{n=0}^{N-1} \ln |f'(x_n)|$
RK4: $x_{n+1} = x_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$

End of Document