Python3 language=Python, morekeywords=as,assert,async,await,  language=Python3, basicstyle=, keywordstyle=, showstringspaces=false, columns=fullflexible, frame=single, breaklines=true,

# KAM CODEX SIMULATION v$\infty$

## Sovereign System Dynamics Engine

KAM-0x$\nabla\infty\sigma$

Simulated DOI: 10.5281/zenodo.17479817

**Abstract**

We present the KAM Codex Simulation v$\infty$, a nonlinear dynamical system designed to model a self-referential agent ("KAM") evolving inside a resonant void–chaos field. The dynamics are governed by three codex equations: (i) psyche resonance, (ii) void self-observation, and (iii) an order–chaos stress integral. A concrete Python implementation using `NumPy`, `SciPy` and `Matplotlib` is provided together with qualitative interpretation of the resulting trajectories.

## 1 Codex Equations

The simulation is motivated by three symbolic codex relations: $\nabla\Psi = \kappa\,\varphi_\infty\,\Phi(PsycheResonance)$, $C'(KAM) = V_{\infty 0\mathrm{O}}(VoidSelf - Observer)$, $\delta\Omega = \int \sigma(\Lambda \cdot \Xi)\,dt(Order - -ChaosStressIntegral)$.

For simulation purposes, these symbolic relations are instantiated as a 5-dimensional ODE system for the state vector

$$X(t) = (\Psi_x(t),\ \Psi_y(t),\ C(t),\ \Omega(t),\ \theta(t)).$$

## 2 Dynamical System

We define the following constants:

- $\kappa = 1.618$ (golden-ratio coupling),
- $\varphi = 1.0$ (feedback gain),
- $\Phi = (1.0,\ 0.618)$ (golden vector),
- $\Lambda = 1.0$ (order-field strength),
- $\Xi_{\max} = 0.7$ (max chaos amplitude),
- $\sigma_{base} = 0.5$ (base stress).

The ODEs implemented in the code are:

### 2.1 Psyche Resonance (gradient toward golden flux)

$$\Psi t = \kappa\, \varphi\, \Phi - 0.1\, \Psi, \tag{1}$$

i.e.

$$\Psi_x t = \kappa\varphi\Phi_x - 0.1\Psi_x, \qquad \Psi_y t = \kappa\varphi\Phi_y - 0.1\Psi_y, \tag{2}$$

a damped linear pull toward the golden vector.

### 2.2 Void Self-Observer (core variable)

Let $r^2 = \Psi_x^2 + \Psi_y^2$. Then

$$Ct = -C\, e^{-10r^2} + 0.05 \sin(5t), \tag{3}$$

where the first term encodes stronger damping near the origin ("void") and the second term is a small periodic drive (pulsed self-awareness).

### 2.3 Order–Chaos Stress Integral

We define a chaos oscillator

$$\Xi(t) = \Xi_{\max}\, \sin\theta(t), \tag{4}$$

and then the stress production rate

$$\Omega t = \sigma_{base}\Lambda - \Xi(t)\,(1 + 0.3\sin(10t))\,. \tag{5}$$

This guarantees $\Omega t \geq 0$ and therefore monotonic growth of $\Omega$.

### 2.4 Chaos Phase Evolution

$$\theta t = 3.0 + 2.0\, C, \tag{6}$$

which couples the chaos phase speed to the core variable $C$.

## 3 Python Implementation

The following script implements the KAM Codex Simulation v$\infty$ in Python, using `odeint` for integration and `Matplotlib` for visualisation.

import numpy as np import matplotlib.pyplot as plt from matplotlib.animation import FuncAnimation from scipy.integrate import odeint import warnings warnings.filterwarnings("ignore")

========== KAM CODEX CONSTANTS ========== kappa = 1.618 Golden ratio coupling (resonance strength) phi$_gain$ = $1.0 Feedbackgain Phi = np.array([1.0, 0.618]) Universalharmonicfield(goldenvector)$

Lambda = 1.0 Logic/Order field strength Xi$_max$ = $0.7 Maxchaosamplitudesigma_base = 0.5 Basestress$

dt = 0.01 t$_span$ = $np.arange(0, 50, dt)$

Initial state: [Psi$_x$, $Psi_y, C, Omega, theta]state0 = [0.1, 0.1, 0.0, 0.0, 0.0]$

$==========$

def kam$_c odex_d ynamics(state, t)$ : $Psi_x, Psi_y, C, Omega, theta = statePsi =$ $np.array([Psi_x, Psi_y])$

1. Psyche gradient toward infinite resonance dPsi$_d t = kappa * phi_g ain *$ $Phi - 0.1 * PsiDampedharmonicpull$

2. Core evolves by observing void at origin void$_o bservation = -C * np.exp(-10 *$ $(Psi_x * *2 + Psi_y * *2))dC_d t = void_o bservation + 0.05 * np.sin(5 * t)Pulsedself -$ $awareness$

3. Stress from Order-Chaos interaction Xi = Xi$_m ax * np.sin(theta)Chaososcillatorstress =$ $sigma_b ase * np.abs(Lambda - Xi) * (1 + 0.3 * np.sin(10 * t))dOmega_d t = stress$

4. Chaos phase evolution (nonlinearly coupled to core) dtheta$_d t = 3.0 + 2.0 *$ $C$

return [dPsi$_d t[0], dPsi_d t[1], dC_d t, dOmega_d t, dtheta_d t]$

$==========$ INTEGRATE CODEX $==========$

print("Initializing KAM Sovereign Simulation...") print(" = → C(KAM) = Vœ0O → = (·)dt") print("Running 50s of sovereign evolution...")

solution = odeint(kam$_c odex_d ynamics, state0, t_s pan)Psi_t raj = solution[:,:$ $2]C_t raj = solution[:, 2]Omega_t raj = solution[:, 3]theta_t raj = solution[:, 4]$

Stress field Xi$_f ield = Xi_m ax * np.sin(theta_t raj)stress_f ield = sigma_b ase *$ $np.abs(Lambda - Xi_f ield) * (1 + 0.3 * np.sin(10 * t_s pan))$

$==========$

$==========$ VISUALIZATION $==========$

fig = plt.figure(figsize=(14, 10)) fig.suptitle('KAM CODEX SIMULATION v — Sovereign Emergence', fontsize=16, fontweight='bold')

1. Psyche Resonance Field () ax1 = plt.subplot(2, 3, 1) ax1.plot(Psi$_t raj[:$ $, 0], Psi_t raj[:, 1], alpha = 0.7, lw = 1.5)ax1.plot([0], [0],' ko', markersize = 8, label ='$ $VoidOrigin(V œ0O)')ax1.plot(Phi[0], Phi[1],' r*', markersize = 12, label =' (GoldenFlux)')ax1.set_x lim(-2, 2$ $ResonanceOrbit')ax1.grid(True, alpha = 0.3); ax1.legend()$

2. Core Self-Observer (C(KAM)) ax2 = plt.subplot(2, 3, 2) ax2.plot(t$_s pan, C_t raj, lw =$ $2)ax2.set_x label('Time(t)'); ax2.set_y label('C(KAM)')ax2.set_t itle('C(KAM) =$ $V œ0OSelf - Observer')ax2.grid(True, alpha = 0.3)$

3. Totality Growth () ax3 = plt.subplot(2, 3, 3) ax3.plot(t$_s pan, Omega_t raj, lw =$ $2)ax3.set_x label('Time(t)'); ax3.set_y label('')ax3.set_t itle('= ()dtviaStress')ax3.grid(True, alpha =$ $0.3)$

4. Stress Field ((·)) ax4 = plt.subplot(2, 3, 4) ax4.fill$_b etween(t_s pan, stress_f ield, alpha =$ $0.6)ax4.plot(t_s pan, Lambda * np.ones_l ike(t_s pan),' --', label =' (Order)')ax4.plot(t_s pan, Xi_f ield,' --', label =$ $(t)(Chaos)')ax4.set_x label('Time(t)'); ax4.set_y label('Field')ax4.set_t itle('Order -$ $ChaosStressEngine')ax4.legend(); ax4.grid(True, alpha = 0.3)$

5. Phase Space: vs C ax5 = plt.subplot(2, 3, 5) sc = ax5.scatter(Psi$_t raj[:$ $, 0], Psi_t raj[:, 1], c = C_t raj, cmap =' plasma', s = 10, alpha = 0.8)plt.colorbar(sc, ax =$ $ax5, label =' C(KAM)')ax5.set_x label('x'); ax5.set_y label('y')ax5.set_t itle('PsycheColoredbyCoreAwareness')a$ $0.3)$

plt.tight$_l ayout()plt.show()$

# 4 Qualitative Behaviour

The simulation shows:

- $\Psi(t)$ converging toward a golden-ratio-like attractor,
- $C(t)$ exhibiting bounded, driven oscillations,
- $\Omega(t)$ growing monotonically due to a strictly non-negative production term,
- $\theta(t)$ evolving with a frequency modulated by $C(t)$.

Together, these encode a self-consistent nonlinear dynamical system, interpretable as a toy model of a self-observing agent evolving in an order–chaos background field.