# Phase 5.12 — Emergent Structures

In this phase, I examine whether the ψ field supports self-organized, localized “lumps” that persist over time. These may resemble solitons or self-trapped wells in other nonlinear field theories. The central question: can ψ-gravity dynamics spontaneously generate and sustain coherent structures from initial fluctuations?

## Core ψ-Gravity Equation (Reminder)

Plain text:  
Gravity(x) = (∇²[space(x) + current(x)²]) × ψ(x)

Force derived as:

Plain text:  
Force(x) = −∇[Gravity(x)]

## Mechanism of Structure Formation

The ψ field is shaped by feedback between:  
- The background curvature term (∇²[space + current²])  
- The local amplitude of ψ(x)

A localized enhancement in ψ strengthens Gravity(x), which modifies the force landscape, potentially reinforcing confinement. If this feedback is positive but bounded, ψ “lumps” form. If it is unstable, ψ collapses or disperses.

## Ansatz for a ψ Lump

I consider an initial localized Gaussian ψ lump:

Plain text:  
ψ(x,0) = A exp(−|x − x₀|² / σ²)

Here:  
- A: amplitude  
- σ: width  
- x₀: center

I evolve this under ψ-gravity dynamics and track whether the lump persists.

## Effective Energy Balance

The energy density is:

For a lump to be stable, the gradient energy (which spreads ψ) must balance the gravitational feedback term (which confines ψ).

## Numerical Simulation: ψ Lump Evolution in 2D

Here I simulate an initial Gaussian lump and test whether it remains localized under ψ-gravity dynamics.

# -----------------------------  
# simulations/phase5\_part5\_12\_ψ-Emergent\_Structures.py  
# Phase 5.12 — ψ lumps and emergent structures  
# -----------------------------  
  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Grid setup  
N = 120  
x = np.linspace(0, 2\*np.pi, N)  
y = np.linspace(0, 2\*np.pi, N)  
X, Y = np.meshgrid(x, y)  
  
# Initial localized ψ lump (Gaussian)  
A = 1.0  
sigma = 0.4  
psi = A \* np.exp(-((X-np.pi)\*\*2 + (Y-np.pi)\*\*2) / sigma\*\*2)  
  
# Define space(x,y) and current(x,y)  
space = np.sin(X) \* np.cos(Y)  
current = np.cos(X) \* np.sin(Y)  
  
# Laplacian operator  
def laplacian(Z, dx):  
 return (  
 -4\*Z  
 + np.roll(Z, 1, axis=0) + np.roll(Z, -1, axis=0)  
 + np.roll(Z, 1, axis=1) + np.roll(Z, -1, axis=1)  
 ) / dx\*\*2  
  
dx = x[1] - x[0]  
  
# Gravity field  
gravity = laplacian(space + current\*\*2, dx) \* psi  
  
# Energy density  
grad\_x, grad\_y = np.gradient(psi, dx, dx)  
energy = 0.5\*(grad\_x\*\*2 + grad\_y\*\*2) + 0.5\*gravity\*psi  
  
# Visualization  
fig, axs = plt.subplots(1, 3, figsize=(15,5))  
axs[0].imshow(psi, extent=[0,2\*np.pi,0,2\*np.pi], origin='lower')  
axs[0].set\_title("Initial ψ Lump")  
  
axs[1].imshow(gravity, extent=[0,2\*np.pi,0,2\*np.pi], origin='lower')  
axs[1].set\_title("Gravity Response")  
  
axs[2].imshow(energy, extent=[0,2\*np.pi,0,2\*np.pi], origin='lower')  
axs[2].set\_title("Energy Density")  
  
plt.show()