Phase 13B – ψ Unification & Field-Theoretic Embeddings  
Part 4: Simulation of ψ Unification Scenarios and Effective Dynamics

Goal  
I now simulate ψ unification dynamics across multiple energy scales, embedding ψ into coupled gravitational, gauge, matter, and dark-sector interactions. The aim is to explore emergent behaviors, fixed points, and effective couplings that ψ mediates. This simulation will serve as a consistency check for ψ-gravity’s predictive unification framework.

Effective Action Recall  
From Part 3, the unified action is:

Plain text:  
S = ∫ d⁴x √(-g) [ 1/2(∂μψ)(∂^μψ) − V(ψ) + λψ∇²(space+current²) + (ξ/2)ψ²R + gψψφ̄φ + gχψχ̄χ − 1/4FμνF^μν − (κ/4)ψFμνF^μν + ηψ²V(φ) ]

Simulation Strategy

* Define ψ potential
* Assign initial conditions at a UV scale (Planck-like) and run RG-like flow toward IR.
* Track effective couplings as ψ evolves.
* Identify emergent fixed points where couplings stabilize.
* Simulate how ψ mediates unification across matter, gauge, and dark fields.

ψ Evolution Equation  
From the Euler–Lagrange equation:

Plain text:  
□ψ + dV/dψ − λ∇²(space+current²) − ξψR − gψφ̄φ − gχχ̄χ − (κ/4)FμνF^μν − 2ηψV(φ) = 0

This governs ψ dynamics with backreaction from all sectors.

Coupling Flow Equations  
Effective couplings run with energy scale μ. Approximate one-loop RG-type forms:

Plain text:  
μ dgψ/dμ = a₁ gψ³ − b₁ gψ ξ  
μ dgχ/dμ = a₂ gχ³ − b₂ gχ η  
μ dξ/dμ = c₁ ξ² − c₂ κ²  
μ dκ/dμ = d₁ κ³ − d₂ gψ gχ  
μ dη/dμ = e₁ η² − e₂ ξ η

Python Simulation Prototype

# simulations/phase13B\_part4\_unification\_flow.py  
import numpy as np  
from scipy.integrate import solve\_ivp  
import matplotlib.pyplot as plt  
  
# Define constants  
a1, b1, a2, b2, c1, c2, d1, d2, e1, e2 = [0.1, 0.05, 0.1, 0.05, 0.02, 0.01, 0.1, 0.05, 0.02, 0.01]  
  
# RG flow equations  
def flow(mu, y):  
 gpsi, gchi, xi, kappa, eta = y  
 dgpsi = a1\*gpsi\*\*3 - b1\*gpsi\*xi  
 dgchi = a2\*gchi\*\*3 - b2\*gchi\*eta  
 dxi = c1\*xi\*\*2 - c2\*kappa\*\*2  
 dkappa= d1\*kappa\*\*3 - d2\*gpsi\*gchi  
 deta = e1\*eta\*\*2 - e2\*xi\*eta  
 # Return derivatives with respect to mu (i.e., dg/dmu). solve\_ivp expects dy/dx.  
 return [dgpsi/mu, dgchi/mu, dxi/mu, dkappa/mu, deta/mu]  
  
# Initial conditions at high scale  
y0 = [0.3, 0.2, 0.1, 0.05, 0.1]  
mu\_span = (1e2, 1e-2) # UV to IR  
sol = solve\_ivp(flow, mu\_span, y0, dense\_output=True)  
  
# Plot flows  
mu\_vals = np.logspace(2, -2, 400)  
y\_vals = sol.sol(mu\_vals)  
  
plt.figure()  
labels = ["gψ", "gχ", "ξ", "κ", "η"]  
for i in range(5):  
 plt.plot(mu\_vals, y\_vals[i], label=labels[i])  
plt.xscale("log")  
plt.xlabel("Energy scale μ")  
plt.ylabel("Coupling strength")  
plt.legend()  
plt.title("ψ Unification Coupling Flow")  
plt.show()