Phase 8 – Part 5  
ψ-Driven Metric Perturbations and Effective Geodesics

🎯 **Goal**  
The aim here is to extend the ψ-gravity framework into a metric-driven description, where ψ not only multiplies curvature but also perturbs the effective spacetime metric.  
This part explores how ψ(x, t) modifies particle geodesics, connecting the desert analogy (ψ = desert floor) with metric deformations that guide motion.

🔧 **Setup**

Core equation (from upgraded ψ-gravity theory):

Plain text:  
Gravity(x) = ( ∇² [ space(x) + current(x)² ] ) \* ψ(x, t)

Force definition:

Plain text:  
F(x, t) = - ∇[ Gravity(x, t) ]

Effective metric perturbation:

Plain text:  
g\_mu,nu(x,t) = η\_mu,nu + α \* ψ(x,t) \* h\_mu,nu(x,t)

Where:

* η\_μν: background Minkowski metric.
* h\_μν(x,t): perturbation kernel derived from space(x) + current(x)².
* α: coupling constant controlling ψ’s strength in geometric embedding.

🌊 **Desert Analogy Extension**

* ψ = desert floor → now acts as the shape of the sand’s curvature.
* Metric = dune geometry written into the landscape.
* Force = dune slope guiding the fish/particle (geodesic-like motion).
* Effective geodesic = the path that minimizes effort across dunes.

📐 **Equations of Motion**

Geodesic equation:

Plain text:  
d²x^μ/dτ² + Γ^μ\_(νρ)(x,t) (dxν/dτ)(dxρ/dτ) = 0

Christoffel symbols (from ψ-perturbed metric):

Plain text:  
Γ^μ\_(νρ) = (1/2) g^μλ ( ∂\_ν g\_λρ + ∂\_ρ g\_λν - ∂\_λ g\_νρ )

🔬 **Sample Case: 2D ψ Field**

Gaussian trench:

Plain text:  
ψ(x,y) = exp(-(x² + y²) / (2σ²))

Effective 2D metric:

Plain text:  
g\_ij(x,y) = δ\_ij + α \* ψ(x,y) \* h\_ij(x,y)

🖥️ **Python Simulation**

# simulations/phase8\_part5\_metric\_geodesics.py

import numpy as np

import matplotlib.pyplot as plt

# Parameters

sigma = 1.0

alpha = 0.5

N = 100

L = 4.0

x = np.linspace(-L, L, N)

y = np.linspace(-L, L, N)

X, Y = np.meshgrid(x, y)

# ψ Gaussian trench

psi = np.exp(-(X\*\*2 + Y\*\*2) / (2 \* sigma\*\*2))

# Simple h\_ij kernel: identity scaled by Gaussian curvature

h\_xx = np.ones\_like(psi)

h\_yy = np.ones\_like(psi)

h\_xy = np.zeros\_like(psi)

# Effective metric components

g\_xx = 1 + alpha \* psi \* h\_xx

g\_yy = 1 + alpha \* psi \* h\_yy

g\_xy = alpha \* psi \* h\_xy

# Determinant of metric (2D subspace)

det\_g = g\_xx \* g\_yy - g\_xy\*\*2

# Visualization: metric deformation strength

plt.figure(figsize=(6,5))

plt.contourf(X, Y, np.sqrt(det\_g), levels=50, cmap="viridis")

plt.colorbar(label="Determinant(g\_ij)")

plt.title("Effective ψ-Metric Determinant (2D)")

plt.xlabel("x")

plt.ylabel("y")

plt.axis("equal")

plt.show()