import numpy as np

import matplotlib.pyplot as plt

# 🛠️ Bell-shaped curvature field

def generate\_bell\_universe(grid\_size=512, height=1.0, steepness=3.0, cold\_lat=-70.0, cold\_lon=180.0):

    lat = np.linspace(-90, 90, grid\_size)

    lon = np.linspace(-180, 180, grid\_size)

    Lon, Lat = np.meshgrid(lon, lat)

    # Convert to radians

    lat\_rad = np.radians(Lat)

    lon\_rad = np.radians(Lon)

    cold\_lat\_rad = np.radians(cold\_lat)

    cold\_lon\_rad = np.radians(cold\_lon)

    # Radial distance from cold spot

    r = np.sqrt((lat\_rad - cold\_lat\_rad)\*\*2 + (lon\_rad - cold\_lon\_rad)\*\*2) + 1e-5

    # Bell-shaped curvature: time compression at base

    curvature = height / (1 + steepness \* r\*\*2)

    curvature /= np.max(curvature)

    return curvature, Lat, Lon

# 🎨 Plot curvature field

def plot\_bell\_universe(curvature, Lat, Lon, tag="Bell Universe Curvature"):

    plt.figure(figsize=(10, 5))

    cf = plt.contourf(Lon, Lat, curvature, levels=50, cmap='coolwarm')

    plt.colorbar(cf, label='Normalized Time Compression')

    plt.title(tag)

    plt.xlabel("Longitude (°)")

    plt.ylabel("Latitude (°)")

    plt.grid(True, alpha=0.3)

    plt.tight\_layout()

    plt.show()

# 🚀 Run simulation

curv, Lat, Lon = generate\_bell\_universe()

plot\_bell\_universe(curv, Lat, Lon)

Right so I’m searching for the cold spot to begin with ok