import healpy as hp

import numpy as np

import matplotlib.pyplot as plt

from scipy.stats import pearsonr

# Load CMB map (uploaded FITS file)

cmb\_map = hp.read\_map("COM\_CMB\_IQU-smica\_2048\_R3.00\_oe2.fits")

nside = hp.get\_nside(cmb\_map)

npix = hp.nside2npix(nside)

# CMB cold spot coordinates (l = 209°, b = -56°)

theta\_cmb = np.radians(90 + 56)

phi\_cmb = np.radians(209)

# Parameters

num\_rings = 10

max\_radius\_deg = 50

num\_trials = 1000

# Ring profile extraction

def ring\_profile(field, theta, phi, nside, npix):

    vec = hp.ang2vec(theta, phi)

    vecs = np.array(hp.pix2vec(nside, np.arange(npix)))

    dots = vecs[0]\*vec[0] + vecs[1]\*vec[1] + vecs[2]\*vec[2]

    dots = np.clip(dots, -1.0, 1.0)

    angs\_deg = np.degrees(np.arccos(dots))

    edges = np.linspace(0, max\_radius\_deg, num\_rings + 1)

    profile = []

    for i in range(num\_rings):

        mask = (angs\_deg >= edges[i]) & (angs\_deg < edges[i+1])

        profile.append(np.mean(field[mask]))

    return profile

# Reference CMB texture profile

cmb\_profile = ring\_profile(cmb\_map, theta\_cmb, phi\_cmb, nside, npix)

# Monte Carlo MBT curvature rotations

correlations = []

for \_ in range(num\_trials):

    rand\_theta = np.radians(np.random.uniform(0, 180))

    rand\_phi = np.radians(np.random.uniform(0, 360))

    vec = hp.ang2vec(rand\_theta, rand\_phi)

    mbt\_field = np.zeros(npix)

    mbt\_field[hp.query\_disc(nside, vec, np.radians(10.0))] = 1.0

    mbt\_profile = ring\_profile(mbt\_field, theta\_cmb, phi\_cmb, nside, npix)

    r, \_ = pearsonr(cmb\_profile, mbt\_profile)

    correlations.append(r)

# Plot histogram

plt.figure(figsize=(8, 5))

plt.hist(correlations, bins=50, color='mediumslateblue', edgecolor='black')

plt.axvline(-0.98, color='red', linestyle='--', label='Your Original Correlation (–0.98)')

plt.xlabel("Pearson Correlation")

plt.ylabel("Number of Random MBT Fields")

plt.title("MBT–SMICA Ring Profile Correlation (Monte Carlo)")

plt.legend()

plt.grid(True, alpha=0.3)

plt.tight\_layout()

plt.show()

# Tail stat

extreme\_hits = np.sum(np.array(correlations) <= -0.98)

p\_value = extreme\_hits / num\_trials

print(f"Fraction of MBT rotations with r ≤ −0.98: {p\_value:.4f} ({extreme\_hits} out of {num\_trials})")

I then did this