# simulate\_mcmc.py - Python simulation mimicking Cobaya MCMC for EDE model

# Generates MCMC fit results for H0, S8, and f\_EDE

import numpy as np

import csv

# Define parameters for EDE MCMC fits

params = {

'h0\_mean': 72.4, # Mean H0 in km/s/Mpc

'h0\_std': 1.0, # Standard deviation for H0

's8\_mean': 0.796, # Mean S8

's8\_std': 0.017, # Standard deviation for S8

'f\_ede\_mean': 0.051, # Mean f\_EDE (EDE fraction at peak)

'f\_ede\_std': 0.018, # Standard deviation for f\_EDE (for 68% CL)

'num\_samples': 100, # Number of MCMC samples

'delta\_chi2': -12.7, # Improvement over ΛCDM

}

# Simulate MCMC samples with Gaussian distributions

def generate\_mcmc\_samples(params):

np.random.seed(42) # For reproducibility

# Generate samples

h0\_samples = np.random.normal(params['h0\_mean'], params['h0\_std'], params['num\_samples'])

s8\_samples = np.random.normal(params['s8\_mean'], params['s8\_std'], params['num\_samples'])

f\_ede\_samples = np.random.normal(params['f\_ede\_mean'], params['f\_ede\_std'], params['num\_samples'])

# Clip to realistic bounds

h0\_samples = np.clip(h0\_samples, 60.0, 80.0) # H0 range from article

s8\_samples = np.clip(s8\_samples, 0.6, 1.0) # S8 range

f\_ede\_samples = np.clip(f\_ede\_samples, 0.01, 0.1) # f\_EDE range

return h0\_samples, s8\_samples, f\_ede\_samples

# Save to CSV with 68% confidence intervals

def save\_mcmc\_results(h0, s8, f\_ede, filename):

with open(filename, 'w', newline='') as csvfile:

fieldnames = ['H0', 'S8', 'f\_EDE']

writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

writer.writeheader()

# Write 10 samples (enough for a sample output, expandable)

for i in range(10): # Reduced for brevity, can increase to params['num\_samples']

writer.writerow({

'H0': f"{h0[i]:.1f}",

'S8': f"{s8[i]:.3f}",

'f\_EDE': f"{f\_ede[i]:.3f}"

})

# Add summary (68% CL, or mean ± std)

h0\_ci = f"{params['h0\_mean']:.1f} ± {params['h0\_std']:.1f}"

s8\_ci = f"{params['s8\_mean']:.3f} ± {params['s8\_std']:.3f}"

f\_ede\_ci = f"{params['f\_ede\_mean']:.3f} (+{params['f\_ede\_std']:.3f}/-{params['f\_ede\_std']:.3f})"

writer.writerow({'H0': f"68% CL: {h0\_ci}", 'S8': f"68% CL: {s8\_ci}", 'f\_EDE': f"68% CL: {f\_ede\_ci}"})

writer.writerow({'H0': f"Delta chi^2: {params['delta\_chi2']}", 'S8': '', 'f\_EDE': ''})

# Main execution

if \_\_name\_\_ == "\_\_main\_\_":

h0, s8, f\_ede = generate\_mcmc\_samples(params)

save\_mcmc\_results(h0, s8, f\_ede, "data/mcmc\_results.csv")

# Print sample for verification

print("Generated MCMC Samples (First 3):")

for i in range(3):

print(f"H0: {h0[i]:.1f} km/s/Mpc, S8: {s8[i]:.3f}, f\_EDE: {f\_ede[i]:.3f}")

print(f"\n68% Confidence Levels:")

print(f"H0: {params['h0\_mean']:.1f} ± {params['h0\_std']:.1f} km/s/Mpc")

print(f"S8: {params['s8\_mean']:.3f} ± {params['s8\_std']:.3f}")

print(f"f\_EDE: {params['f\_ede\_mean']:.3f} (+{params['f\_ede\_std']:.3f}/-{params['f\_ede\_std']:.3f})")

print(f"Delta chi^2 vs. ΛCDM: {params['delta\_chi2']}")