import numpy as np

def hermitian(X):

# エルミート共役（随伴）：複素共役の転置

return X.conj().T

def ndist\_matrix(nc):

# 平均0, 分散1の正規分布に従う実数値行列

return np.random.normal(0.0, 1.0, size=(nc, nc))

def generate\_lie\_su(nc):

# 反エルミート化してからトレースを落として su(n) の元にする

tmp = 0.5 \* (ndist\_matrix(nc) + 1j \* ndist\_matrix(nc))

tmp = tmp - hermitian(tmp) # 反エルミート化

tmp = tmp - (np.trace(tmp) / nc) \* np.eye(nc, dtype=complex) # traceless 化

return tmp

def generate\_lie\_sl(nc):

# 複素行列を作ってからトレースを落として sl(n, C) の元にする

tmp = ndist\_matrix(nc) + 1j \* ndist\_matrix(nc)

tmp = tmp - (np.trace(tmp) / nc) \* np.eye(nc, dtype=complex) # traceless 化

return tmp