```
import numpy as np
import matplotlib.pyplot as plt

A9 = np.array([
    [-2, -1,  1,   0],
    [-1, -2,  0,   1],
    [-2, -1,  0, -31j],
    [-1, -2, 31j,  0]
], dtype=complex)

def compute_block_numerical_range(matrix, block_sizes, num_samples=150000):
    n = matrix.shape[0]
    points = []
    cum_sizes = np.insert(np.cumsum(block_sizes), 0, 0)
    for _ in range(num_samples):
        parts = []
        for i in range(len(block_sizes)):
            size = block_sizes[i]
            z_part = np.random.randn(size) + 1j * np.random.randn(size)
            z_part /= np.linalg.norm(z_part)
            parts.append(z_part)
        V = np.zeros((n, len(block_sizes)), dtype=complex)
        for i in range(len(block_sizes)):
            V[cum_sizes[i]:cum_sizes[i+1], i] = parts[i]
        A_compressed = np.dot(np.conj(V).T, np.dot(matrix, V))
        eigvals = np.linalg.eigvals(A_compressed)
        points.extend(eigvals)
    return np.array(points)

print("Calcul des structures cubiques... (génération des figures)")

points_A9 = compute_block_numerical_range(A9, [1, 1, 2], num_samples=200000)
vals_A9 = np.linalg.eigvals(A9)

fig, (ax) = plt.subplots(1, 1, figsize=(14, 6))
ax.scatter(points_A9.real, points_A9.imag, s=0.02, c='red', alpha=0.3)
ax.scatter(vals_A9.real, vals_A9.imag, color='black', s=25, zorder=5)
ax.set_title(r"Cubic Range de $\mathcal{A}_9$ (Croissant et antenne)")
#ax.set_xlim([-4, 4])
#ax.set_ylim([-5, 5])
ax.set_aspect('equal')
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
```