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

# Function to calculate the channel capacity for a given SNR and
eigenvalue
def channel_capacity(SNR, eigenvalue, NT, NR, q):
    capacity = NT * (np.log2(1 + SNR * eigenvalue / NR) - np.log2(np.e) *
(2 * q * (NT - NR) / (NR * eigenvalue)))
    return capacity

# Function to calculate the ergodic capacity for a given SNR and
eigenvalues
def ergodic_capacity(SNR, eigenvalues, NR):
    capacity = np.mean(np.maximum(np.log2(1 + SNR * eigenvalues / NR), 0))
    return capacity

# Set parameters
NT = 2   # Number of transmit antennas
NR = 2   # Number of receive antennas
q = 1.5  # Shape parameter for Hoyt fading

# Generate random eigenvalues for demonstration
num_samples = 100
eigenvalues = np.random.uniform(low=0.1, high=1.0, size=num_samples)

# Set SNR values
SNR = np.linspace(0, 20, 100)  # Signal-to-Noise Ratio range

# Calculate channel capacity for the given SNR and eigenvalue
channel_cap = np.zeros_like(SNR)
for i in range(len(SNR)):
    channel_cap[i] = channel_capacity(SNR[i], eigenvalues[0], NT, NR, q)

# Calculate ergodic capacity for the given SNR and eigenvalues
erg_cap = np.zeros_like(SNR)
for i in range(len(SNR)):
    erg_cap[i] = ergodic_capacity(SNR[i], eigenvalues, NR)

# Plotting
plt.plot(SNR, channel_cap, label='Channel Capacity')
plt.plot(SNR, erg_cap, label='Ergodic Capacity')
plt.xlabel('SNR (dB)')
plt.ylabel('Capacity')
plt.title('Channel Capacity vs Ergodic Capacity')
plt.legend()
```

```
plt.grid(True)
plt.show()
```