GMC - ChatGPT

From Srinivasan Kannan <ka.shrinivaasan@gmail.com>
To Krishna iResearch - Srinivasan Kannan (also known as: Ka.Shrinivaasan, Shrinivas Kannan) <ksrinivasan@krishna-iresearch.org>, Srinivasan Kannan <ka.shreenivaasan@gmail.com>
Date 2025-12-23 10:08

Below is a **clean, minimal Python example** of **Gaussian Multiplicative Chaos (GMC)** in 1D and 2D, following the standard construction used in probability and statistical physics.

1️⃣ Concept (brief)

Gaussian Multiplicative Chaos is defined as

$$M_\gamma(dx) = \lim_{\varepsilon \to 0} \exp\Big(\gamma X_\varepsilon(x) - \frac{\gamma^2}{2}\mathbb{E}[X_\varepsilon(x)^2]\Big)\,dx$$

where:

- is a **log-correlated Gaussian field**
- (subcritical regime)
- is a mollified (regularized) field

2️⃣ 1D Gaussian Multiplicative Chaos (log-correlated field)

🔹 Step-by-step numerical approximation

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

# Parameters
N = 4096           # grid size
gamma = 0.7        # GMC parameter (gamma < sqrt(2))
L = 1.0            # domain length

# Spatial grid
x = np.linspace(0, L, N)

# Fourier frequencies
k = np.fft.fftfreq(N, d=L/N)
k[0] = 1.0  # avoid division by zero

# Generate log-correlated Gaussian field
random_phase = np.random.normal(size=N) + 1j*np.random.normal(size=N)
X_hat = random_phase / np.sqrt(np.abs(k))
X = np.real(np.fft.ifft(X_hat))

# Renormalization
var_X = np.var(X)
gmc_measure = np.exp(gamma * X - 0.5 * gamma**2 * var_X)

# Normalize as a probability measure
gmc_measure /= np.sum(gmc_measure)

# Plot
plt.plot(x, gmc_measure)
plt.title("1D Gaussian Multiplicative Chaos")
plt.xlabel("x")
plt.ylabel("My(dx)")
plt.show()
```

✔ What this does

- Constructs a **log-correlated Gaussian field**
- Applies **GMC exponential renormalization**
- Produces a **random multifractal measure**

3️⃣ 2D Gaussian Multiplicative Chaos (more common)

🔹 2D Log-correlated field via FFT

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

# Parameters
N = 512
gamma = 1.2   # must satisfy gamma < 2 in 2D
L = 1.0

# Grid
kx = np.fft.fftfreq(N, d=L/N)
ky = np.fft.fftfreq(N, d=L/N)
kx, ky = np.meshgrid(kx, ky)
k = np.sqrt(kx**2 + ky**2)
k[0, 0] = 1.0

# Gaussian field in Fourier space
noise = np.random.normal(size=(N, N)) + 1j*np.random.normal(size=(N, N))
X_hat = noise / k
X = np.real(np.fft.ifft2(X_hat))

# Renormalization
var_X = np.var(X)
M = np.exp(gamma * X - 0.5 * gamma**2 * var_X)

# Normalize
M /= np.sum(M)

# Plot
plt.imshow(M, cmap="inferno")
plt.colorbar(label="GMC measure")
plt.title("2D Gaussian Multiplicative Chaos")
plt.show()
```

4️⃣ Key Notes (important)

- ✔ This is a **numerical approximation**, not the strict ε→0 limit
- ✔ Renormalization term is **essential**
- ✔ Use **small γ** to avoid degeneration
- ✔ Increasing grid resolution improves convergence

5️⃣ Variations you may want

- Circle-average regularization
- Exact covariance kernel construction
- Critical GMC (γ = √2d)
- Time-dependent GMC
- Application to turbulence / Liouville quantum gravity

If you tell me **your application** (theory, simulation, finance, turbulence, LQG, urban modeling, etc.), I can tailor the code precisely.

NeuronRain Documentation and Licensing: http://neuronrain-documentation.readthedocs.io/en/latest/
Personal website(research): https://acadpdrafts.readthedocs.io/en/latest/
