# PHENOMENOLOGY AND MODELLING OF FINANCIAL MARKETS

## Petite Classe n. 1

### Time Series Simulation and Analysis

ECO 586 - J. Moran
M. Benzaquen

6 January 2020

## Introduction

Through the course and exercise sessions you are expected to encounter and analyse numerical time series from financial data. The goal of this session is to familarize you with the numerical analysis of time series, as well as to give you some basic intuition on what can cause the emergence of certain stylized facts in financial time series.

### Part 1 : Fractional brownian motion (fBM)

A fractional brownian motion is a stochastic process with the property

$$\langle x_H(t)x_H(s)\rangle = \frac{1}{2}(|t|^{2H} + |s|^{2H} - |t-s|^{2H}), \quad t,s > 1\,. \tag{1}$$

The parameter $H$ is called the Hurst exponent and controls the "roughness" of the process. $H = 1/2$ corresponds to standard brownian motion. For $H < 1/2$ the process is "rougher" than standard brownian motion and the increments are anti-correlated (mean-reversion). For $H > 1/2$ the process is "smoother", with trending increments.

1. Let $L$ be a $T \times T$ matrix. Show that the correlation matrix of the vector $Y = LX$, where $X$ is an iid. gaussian vector of size $T$, is given by $C = L^t L$.
2. Define a function `C(i,j,H)` giving the $C_{ij}$ term of the correlation matrix of an fBM with exponent H. Using the function `np.fromfunction()` compute the matrix for an fBM of length T=1000 and the exponent of your choice.
   Use the Cholesky decomposition with `np.linalg.cholesky` to generate a fractional brownian motion, rescaled as
   $$\tilde{x}_H(t) = 1/T^H x_H(t) \tag{2}$$
   so that all simulated curves have roughly the same scale.
3. Group everything into a function `gen_fbm(T,H)` that returns an instance of a (rescaled) fBM with exponent $H$ of length $T$. Plot for $H = 0.25, 0.5, 0.75$. Comments?
4. To generate statistics over the fBM process it is necessary to generate many instances of it. Define now a function `gen_N_fbm(T,N,H)` that returns a $T \times N$ array with $N$ instances of the fractional brownian motion for the same exponent $H$. Hint: what happens if we take the vector $X$ to be gaussian iid matrix of size $T \times N$?
5. Create a DataFrame `df` with 100 columns, each containing a realisation of the fBM with $H = 0.3$ and $T = 1000$.
6. Create a function that computes the value $(x(t) - x(t+\tau))^2$, averages it over $t$ for a given realization, and then averages over all of th realisations in `d` for a single value $\tau$. Use it to compute the variogram,
   $$V(\tau) = \left\langle (x(t+\tau) - x(t))^2 \right\rangle \tag{3}$$
   for $0 \leq \tau < 500$.

7. Plot alongside the theoretical value on a log-log scale, $V_{th}(\tau) \propto \tau^H$. Do the same for $H = 0.75$ and compare.
8. Compute the volatility signature plots for the two fBMs you generated and compare. Comments?

### Part 2 : The Ornstein-Uhlenbeck process

The Ornstein-Uhlenbeck process (centered around 0) is defined by the following SDE:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\omega x + \sigma\xi(t), \quad \langle \xi(t)\xi(t')\rangle = \delta(t-t'). \tag{4}$$

We remind that the discretization according to the Itô prescription of this SDE is given by:

$$x_{(N+1)\Delta t} = x_{N\Delta t}(1-\omega\Delta t) + \sigma\sqrt{\Delta t}\xi_N \tag{5}$$

where the $(\xi_N)$ variables are normally distributed with variance 1.

1. Write a function `ornstein_uhlenbeck(x_0, omega, omega, dt, T)` that returns a realisation of the Ornstein-Uhlenbeck process with initial condition $x_0$ and with parameters $\mu$ and $\omega$ over $T$ steps with a discretization $dt$.
2. To generate statistics over the Ornstein-Uhlenbeck process it is necessary to generate many instances of it. Write a new function `N_Ornstein_uhlenbeck(x_0, sigma, mu, dt, N, T)` that generates a $T \times N$ array where each column is a realisation of the O-U process with corresponding parameters. The $x_0$ initial condition is now a vector of size $N$. We remind that in `Python` `a[i,:]` accesses the i-th row of array `a`, and `a[:,j]` its j-th column.
3. Show that the solution of the Ornstein-Uhlenbeck process is given by

$$x(t) = x(0)e^{-\omega t} + \sigma\int_0^t \mathrm{d}s\; e^{-\omega(t-s)}\xi(s) \tag{6}$$

   (hint: integrate the SDE as a standard ODE).
   Show that $\langle x(t)\rangle \underset{t\to\infty}{\longrightarrow} 0$.
   Take now $x(0) = 0$ for simplicity. Compute the product $x(t)x(t+\tau)$ and average it over $\xi$, obtaining the correlation function:

$$C(t,\tau) = \frac{\sigma^2}{2\omega}\left(e^{-\omega|\tau|} - e^{-\omega(2t+\tau)}\right) \tag{7}$$

   For long times, we therefore have $C(t,\tau) \underset{t\gg 1/\omega}{\longrightarrow} \frac{\sigma^2}{2\omega}e^{-\omega|\tau|}$.
4. Create a DataFrame `df` where each row is a time point and with 100 columns corresponding to an instance of an Ornstein-Uhlenbeck process with $\sigma = 1$ and your choice of $\omega$. (hint: use `pd.DataFrame(x)` where `x` is the output of a function you have defined). Take again $T = 10^5$ and $dt = 10^{-3}$.
5. Create a function that computes the correlation $x(t)x(t+\tau dt)$, averages it over $t$ for a given realisation, and then averages over all the realisations in `df` for a single value $\tau$. Compute this correlation function over a set of points (at least up to $\tau dt = 2$) and compare this with $\sigma/(2\omega)e^{-\omega dt\tau}$.
6. Compute the variogram $V(\tau)$ of the Ornstein-Uhlenbeck process, as well as the volatility signature plot $V(\tau)/\tau$. Compare it to the fBM. What do you notice? Be smart, you already have the correlation function!

### Part 3 : A physical example: turbulent flow

We simulate a physical chaotic system using the Lorenz attractor, an idealized model of a fluid with coordinates $(x, y, z)$, where $x$ is the convective intensity, $y$ the temperature difference between

descending and ascending currents and $z$ the difference in vertical temperature profile. This evolution is given by

$$\begin{aligned} \dot{x} &= \sigma(y-x) \\ \dot{y} &= rx - y - xz \\ \dot{z} &= xy - bz. \end{aligned} \tag{8}$$

We'll simulate this through $10^6$ time steps under a discretisation $dt = 10^{-2}$.

1. Run the code that is given to you to simulate this system. The system jumps between two attractors. Describe the plot of $z_t$.
2. We will interpret $z_t$ as a velocity. Since we are interested in statistical properties, it is better to detrend and rescale this variable $z_t \to (z_t - \langle z_t \rangle)/\langle z_t^2 \rangle_c$. We then define a variable $Z_t = \sum_{l=0}^{t} z_t$, and we plot its evolution. Comments?
3. Create a dataframe `df` containing a column for $Z_t$ called `df['Z']`. Compute the variogram of $Z_t$, for $\tau$ in `np.logspace(0,6,20, dtype=int)`. What do you see?
4. Plot the histogram of
$$\frac{x(t) - x(t+\tau)}{\sigma(\tau)} \tag{9}$$
where
$$\sigma(\tau) := \sqrt{V(\tau)} = \sqrt{\langle (x(t+\tau) - x(t))^2 \rangle}$$
in the same plot for different values of $\tau$. What do you notice?