## The FFT with the numpy.fft module

1. Biến đổi Fourier Rời rạc (DFT) là gì?

Trước khi đi sâu vào FFT, chúng ta cần hiểu DFT. DFT là một phép toán xử lý tín hiệu, chuyển đổi tín hiệu từ miền thời gian (hoặc miền không gian) sang miền tần số. Nói cách khác, nó cho biết tín hiệu được cấu tạo từ những tần số nào và mỗi tần số mạnh yếu ra sao.

Ví dụ, khi phân tích một bản nhạc bằng DFT, ta sẽ biết được bản nhạc đó chứa những nốt nào và cường độ của mỗi nốt.

2. Tại sao lại cần FFT?

DFT chậm: Tính toán DFT trực tiếp theo công thức có độ phức tạp O(N^2), với N là số lượng mẫu trong tín hiệu. Điều này có nghĩa là thời gian tính toán tăng rất nhanh khi tín hiệu lớn.

FFT nhanh hơn: FFT là một thuật toán thông minh giúp tính DFT với độ phức tạp chỉ O(N log N). Nhờ vậy, FFT giúp tính toán DFT cho tín hiệu lớn trở nên khả thi hơn.

3. Ý tưởng chính của FFT

FFT dựa trên chiến lược "chia để trị":

* Chia: Chia bài toán DFT lớn thành các bài toán DFT nhỏ hơn.
* Trị: Giải quyết các bài toán DFT nhỏ một cách đệ quy.
* Kết hợp: Kết hợp kết quả của các bài toán DFT nhỏ để thu được kết quả của bài toán DFT ban đầu.

4. Thuật toán Cooley-Tukey (một thuật toán FFT phổ biến)

Đây là thuật toán FFT phổ biến nhất, hoạt động như sau:

* Chia: Chia tín hiệu thành hai chuỗi con bằng nhau: một chuỗi chứa các mẫu có chỉ số chẵn, một chuỗi chứa các mẫu có chỉ số lẻ.
* Trị: Tính toán DFT của hai chuỗi con một cách đệ quy. Lặp lại bước này cho đến khi mỗi chuỗi con chỉ còn 1 mẫu.
* Kết hợp: Kết hợp DFT của các chuỗi con để thu được DFT của tín hiệu ban đầu. Bước này sử dụng các tính chất toán học của DFT để giảm số lượng phép toán.