FFT (FAST FOURIER TRANSFORM)

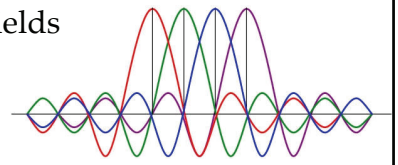
A METHOD FOR THE EFFICIENT COMPUTATION OF THE DFT (DISCRETE FOURIER TRANSFORM)

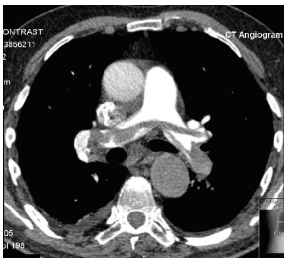
FFT References

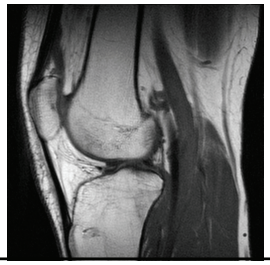
- FFT derivation described in nearly all DSP textbooks (at some level)
  - *Discrete-Time Signal Processing*, Oppenheim and Schafer, Prentice Hall
  - *Theory and Applications of Digital Signal Processing*, Rabiner and Gold
    - Has good hardware discussions and a large number of FFT algorithms with unusual dataflow graphs

FFT Applications

- Widely used throughout many fields
  - Communications
    - Ex: OFDM modulation. e.g., Wi-Fi
  - Medical imaging
    - Ex: X-ray computed tomography (CT)
    - Ex: Magnetic resonance imaging (MRI)
  - Signal analysis
  - Radar



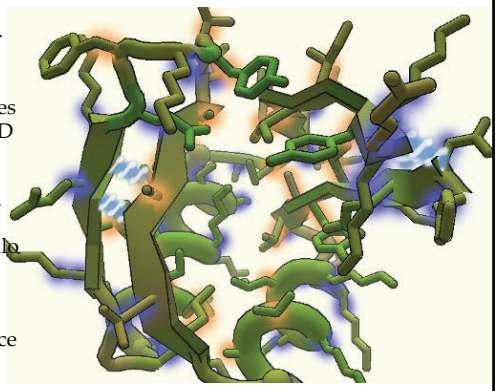




FFT Applications

- Widely used throughout many fields
  - Scientific computing
    - Ex: Car-Parrinello Method for protein folding simulations. "The execution time of Car-Parrinello based first principles calculation is dominated by 3D FFTs of electronic-state vectors…" [T. Sasaki, K. Betsuyaku et al., "Reconfigurable 3D-FFT Processor for the Car-Parrinello Method," 2005]

      Foldit game: http://fold.it/portal/info/science



Discrete Fourier Transform (DFT)

- $X(k)$ is the DFT of the N-point sequence $x(n)$

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-i2\pi nk/N}, k = 0,1,\ldots, N-1$$

or we can write

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, k = 0,1,\ldots, N-1$$

with

$$W_N = e^{-i2\pi/N}$$

$$W_N = \cos\left(\frac{2\pi}{N}\right) - i\sin\left(\frac{2\pi}{N}\right)$$

W_N Coefficient

- W_N is a complex coefficient that is constant for a particular length DFT/FFT
- W_N^{nk} changes
- W_N sometimes called the "N^{th} root of unity"
  - $W_N^N = 1$
- W_N is periodic with N; that is, $W_N^n = W_N^{n+mN}$ for any integer m

Discrete Fourier Transform (DFT)

- Computational complexity
  - Each of the N $X(k)$ outputs requires N (complex) multiplications and N-1 (complex) additions
  - Straightforward DFT requires Order(N^2) calculations

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, k = 0,1,\ldots, N-1$$

Inverse Discrete Fourier Transform (IDFT)

- $x(n)$ is the IDFT of the N-point sequence $X(k)$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{i2\pi nk/N}, n = 0,1,\ldots, N-1$$

or we can write

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk}, n = 0,1,\ldots, N-1$$

Inverse Discrete Fourier Transform (IDFT)

- Computational complexity
  - Each of the N $x(n)$ outputs requires N (complex) multiplications and N-1 (complex) additions
  - Straightforward IDFT also requires Order(N^2) calculations

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k) W_N^{-nk}, n = 0,1,\ldots, N-1$$

Fast Fourier Transform (FFT)

- The FFT is an efficient algorithm for calculating the Discrete Fourier Transform
- Widely credited to Cooley and Tukey (1965)
  - "An Algorithm for the Machine Calculation of Complex Fourier Series," in *Math. of Comput.*, volume 19, April 1965.
- Previous to 1965, nearly all DFTs were calculated using N^2 algorithms
- Cooley, Lewis, and Welch (1967) report some of the earlier known discoverers. They cite a paper by Danielson and Lanczos (1942) describing a type of FFT algorithm and its application to X-ray scattering experiments. The Danielson and Lanczos paper refers to two papers written by Runge (1903; 1905). Those papers and lecture notes by Runge and König (1924), describe two methods to reduce the number of operations required to calculate a DFT: one exploits the *symmetry* and a second exploits the *periodicity* of the DFT kernel $e^{i\theta}$. Exploiting symmetry allows the DFT to be computed more efficiently than a direct process, but only by a constant factor; the algorithm is still $O(N^2)$. By taking advantage of the periodicity of the DFT kernel, much greater gains in efficiency are possible, such that the complexity can be brought below $O(N^2)$. The length of transforms Runge and König worked with were relatively short (since all computation was done by hand); consequently, both methods brought comparable reductions in computational complexity and Runge and König actually emphasized the method exploiting symmetry.

Fast Fourier Transform (FFT)

- Fifteen years after Cooley and Tukey's paper, Heideman et al. (1984), published a paper providing even more insight into the history of the FFT including work going back to Gauss (1866). Gauss' work is believed to date from October or November of 1805 and, amazingly, predates Fourier's seminal work by two years.
- The FFT is order $N \log N$
- As an example of its efficiency, for a one million point DFT:
  - Direct DFT: 1×10^{12} operations
  - FFT: 2×10^{7} operations
  - A speedup of 52,000!

FFT Derivation

- Consult a reference such as Oppenheim and Schafer for details
- Most straightforward way is to separate summation index into odd and even summations, substitute summation index, and recognize DFTs for x_{even} and x_{odd}
- W_N constants or coefficients are called *twiddle factors*