

Fast Fourier Transform

Matt McCarthy

Christopher Newport University

CPSC 621

November 30, 2015

Discrete Fourier Transform

Definition 1 (Discrete Fourier Transform)

Let $\mathbf{X} = (x_0, x_1, \dots, x_{n-1}) \in \mathbb{C}^n$. Then the *Discrete Fourier Transform* of \mathbf{X} is defined as $\mathbf{Y} = (y_0, y_1, \dots, y_{n-1})$ where

$$y_j := \sum_{k=0}^{n-1} x_k \omega^{jk}$$

with $\omega = e^{2\pi i/n}$. Furthermore, we denote $\mathbf{Y} = \mathcal{F}(\mathbf{X})$.

Discrete Fourier Transform

Definition 1 (Discrete Fourier Transform)

Let $\mathbf{X} = (x_0, x_1, \dots, x_{n-1}) \in \mathbb{C}^n$. Then the *Discrete Fourier Transform* of \mathbf{X} is defined as $\mathbf{Y} = (y_0, y_1, \dots, y_{n-1})$ where

$$y_j := \sum_{k=0}^{n-1} x_k \omega^{jk}$$

with $\omega = e^{2\pi i/n}$. Furthermore, we denote $\mathbf{Y} = \mathcal{F}(\mathbf{X})$.

Complexity: $\Theta(n^2)$.

Slightly Faster Fourier Transform

If we assume n is even, then by symbol pushing we get

$$\begin{aligned}y_j &= \sum_{k=0}^{n/2-1} x_{2k} \omega^{(2k)j} + \sum_{k=0}^{n/2-1} x_{2k+1} \omega^{(2k+1)j} \\&= \sum_{k=0}^{n/2-1} x_{2k} e^{2(2\pi i/n)jk} + \sum_{k=0}^{n/2-1} x_{2k+1} \omega^j e^{2(2\pi i/n)jk} \\&= \sum_{k=0}^{n/2-1} x_{2k} \left(e^{2\pi i/n}\right)^{2jk} + \omega^j \sum_{k=0}^{n/2-1} x_{2k+1} \left(e^{2\pi i/n}\right)^{2jk}\end{aligned}$$

Slightly Faster Fourier Transform

If we assume n is even, then by symbol pushing we get

$$\begin{aligned}y_j &= \sum_{k=0}^{n/2-1} x_{2k} \omega^{(2k)j} + \sum_{k=0}^{n/2-1} x_{2k+1} \omega^{(2k+1)j} \\&= \sum_{k=0}^{n/2-1} x_{2k} e^{2(2\pi i/n)jk} + \sum_{k=0}^{n/2-1} x_{2k+1} \omega^j e^{2(2\pi i/n)jk} \\&= \sum_{k=0}^{n/2-1} x_{2k} \left(e^{2\pi i/n}\right)^{2jk} + \omega^j \sum_{k=0}^{n/2-1} x_{2k+1} \left(e^{2\pi i/n}\right)^{2jk}\end{aligned}$$

Let $\tilde{\omega} = \omega^2$ and we have

$$y_j = \sum_{k=0}^{n/2-1} x_{2k} \tilde{\omega}^{jk} + \omega^j \sum_{k=0}^{n/2-1} x_{2k+1} \tilde{\omega}^{jk}.$$

Fast Fourier Transform

If $n = 2^k$ for some $k \in \mathbb{Z}^+$, then we can iterate this process using the following algorithm.

Fast Fourier Transform

If $n = 2^k$ for some $k \in \mathbb{Z}^+$, then we can iterate this process using the following algorithm.

The one-dimensional, unordered, radix 2, FFT algorithm.

```
1: function R-FFT(X,Y, $n,\omega$ )
2:   if  $n=1$  then
3:      $y_0 = x_0$ 
4:   else
5:     Let  $\mathbf{Q} = \mathbf{0}, \mathbf{T} = \mathbf{0} \in \mathbb{C}^n$ 
6:     R-FFT( $(x_0, x_2, \dots, x_{n-2}), (q_0, q_2, \dots, q_{n-2}), n/2, \omega^2$ )
7:     R-FFT( $(x_1, x_3, \dots, x_{n-1}), (t_1, t_3, \dots, t_{n-1}), n/2, \omega^2$ )
8:     for all  $j \in \{0, 1, \dots, n-1\}$  do
9:        $y_j = q_{j \bmod n/2} + \omega^j t_{j \bmod n/2}$ 
10:    end for
11:  end if
12: end function
```

Fast Fourier Transform: Serial Analysis

- Since $n = 2^k$, we do $\lg n = k$ steps
- At the m th level of recursion we do 2^m FFTs of size $n/2^m$
 - Each level is $\Theta(n)$
- Thus, FFT is $\Theta(n \lg n)$.

References I



Author

Title

where

Thank you!