Fourier transforms

Burton Rosenberg

5 December 2003
Revised: 5 December 2008

Overview

This is an introduction to fast fourier transform for Algorithms CS 517. It is intended to be simple, but not too simple. It takes a mathematically sophisticated approach.

Definition of the Fourier transform

Let $\{\, a_j \,|\, j = 0, \ldots, n-1 \,\}$ be n samples of a function. Although these are generally real numbers, they could be complex numbers. The *Fourier Transform* associates to the set of samples a set of complex numbers, $\{\, \hat{a}_k \,|\, k = 0, \ldots, n-1 \,\}$, called the *Fourier coefficients*.

A function can be represented either by its samples or by its fourier coefficients. As samples, it is represented as a weighted sum of functions $\{\, e_j \,|\, j = 0, \ldots, n-1 \,\}$ called the *sample basis*,

$$e_j(k) = \begin{cases} 1 & \text{if } j = k, \\ 0 & \text{else} \end{cases}$$

The function is also represented as a weighted sum of functions $\{\, \chi_k \,|\, k = 0, \ldots, n-1 \,\}$ called the *fourier basis*,

$$\chi_k(j) = e^{2\pi i jk/n} = \cos(2\pi jk/n) + i \sin(2\pi jk/n)$$

The two representations give the same function. Therefore the fourier transform is summarized by the equality,

$$(1/n) \sum_{k=0}^{n-1} \hat{a}_k \chi_k = \sum_{j=0}^{n-1} a_j e_j$$

The scale factor $1/n$ is needed because the functions χ_k of the fourier basis are "bigger" then the e_j in the sample basis. The factor compensates for this.

Each side of the equality is a function. We are representing a function without explicitly showing the free variable; we are writing f instead of $f(x)$. To evaluate the right-hand-side at a particular value k, $k = 0, \ldots, n-1$, we apply the evaluation to each component function and sum,

$$\left. \sum_{j=0}^{n-1} a_j e_j \right|_k = \sum_{j=0}^{n-1} a_j e_j(k) = a_k$$

Hence the name "sample basis", the e_j glue together samples into a function where the weight a_j of e_j is the value of the function sampled at j.

The left-hand-side evaluation gives,

$$\left. (1/n) \sum_{k=0}^{n-1} \hat{a}_k \chi_k \right|_j = (1/n) \sum_{k=0}^{n-1} \hat{a}_k e^{2\pi ijk/n} = (1/n) \sum_{k=0}^{n-1} \hat{a}_k \left(\cos(2\pi jk/n) + i \sin(2\pi jk/n)\right)$$

This could be read as the superposition of overtones of the fundamental (complex) sine wave

$$\cos k\omega j + i \sin k\omega j,$$

with $\omega = 2\pi/n$, the overtones generated by $k = 0, \ldots, n-1$, and j being the point of sampling, $\hat{a}_k$ is the (complex) weight of the k-th overtone in the superposition.

Calculation of the Fourier transform

A function can be represented as a sum of samples in the sample basis, or a superposition of overtones in the fourier basis. Each representation has its strengths. Represented in the sampling basis is natural when a signal is sampled or reconstructed. Represented in the fourier basis is more useful when the function is modified: for instance when a sound signal is equalized or echo-cancelled. To use both representations it is necessary to preform the transformation and its inverse. That is, from the samples a_j to calculate the coefficients $\hat{a}_k$, and from the coefficients $\hat{a}_k$ to calculate the samples a_j.

Given the fourier coefficients, the samples are easy to recover: evaluate both sides of the equation at j,

$$a_j = (1/n) \sum_{k=0}^{n-1} \hat{a}_k e^{2\pi ikj/n}$$

This can be written as a polynomial,

$$f(y) = (1/n) \sum_{k=0}^{n-1} \hat{a}_k y^k$$

and $a_j = f(\omega^j)$ where ω is the principal n-th root of unity $\omega = e^{2\pi i/n}$.

Think of f as a surface in complex space described by a $n-1$ degree polynomial. It is entirely determined by its values on the n-th roots of unity. It is also entirely determined by its n coefficients. The Fourier transform is the passage between these two descriptions. We can think of constructing the surface from the coefficients and then recovering the a_j, or we can think of constraining that the surface pass through the a_j at ω^j and then see what are the resulting values of the coefficients.

It is not as easy, however, to derive the polynomial's coefficients from its values on the roots of unity than it is to simply evaluate a given polynomial at the roots of unity. However, the special structure of the roots of unity gives a strange duality — we can nearly interchange the role of the coefficients and the values of the polynomial at the roots of unity. The polynomial,

$$g(y) = \sum_{j=0}^{n-1} a_j y^j$$

gives the values of $\hat{a}_k$ when evaluated at ω^{-k},

$$\hat{a}_k = g(\omega^{-k})$$

(note carefully the negative sign in the exponent).

We prove this. Since $a_j = f(\omega^j)$, then

$$\begin{aligned} g(\omega^{-k}) &= \sum_{j=0}^{n-1} f(\omega^j)\omega^{-kj} \\ &= (1/n)\sum_{j=0}^{n-1} \omega^{-kj} \sum_{l=0}^{n-1} \hat{a}_l \omega^{jl} \\ &= (1/n)\sum_{l=0}^{n-1} \hat{a}_l \sum_{j=0}^{n-1} \omega^{j(l-k)} \\ &= \hat{a}_k \end{aligned}$$

because the sum of powers of a root of unity is zero, unless the root of unity in question is one,

Lemma 1 *Let $\omega = e^{2\pi i/n}$ be the principal n-th root of unity. Then,*

$$\sum_{j=0}^{n-1} \omega^{kj} = \begin{cases} n & \textit{if } \omega^k = 1 \\ 0 & \textit{else} \end{cases}$$

Proof: Denote the sum by S. $S = \omega^k S$, because we are only causing a renaming of the index, $j' = j + 1$. If $\omega^k \neq 1$, this implies $S = 0$.

Fast implementations, the FFT

We have established that the fourier transform of a sequence of n complex numbers is the evaluation of a polynomial at the n powers of the principal n-th root of unity This can be accomplished simply in time $\Theta(n^2)$, by n evaluations, where each evaluation is done in time $\Theta(n)$ using *Horner's rule*:

$$f(y) = \sum_{j=0}^{n-1} a_j y^j = a_0 + y(a_1 + y(a_2 + y(\ldots + y(a_{n-2} + y a_{n-1})\ldots))$$

the Fast Fourier transform performs all these evaluations in time $\Theta(n \log n)$.

Write f as the sum of even and odd powers, and factor a y from the odd-power sum,

$$f(y) = (a_0 + a_2 y^2 + \ldots) + y(a_1 + a_3 y^2 + \ldots) = f_e(y') + y f_o(y')$$

where $y' = y^2$. Assuming n even, f_o and f_e are both degree $n/2 - 1$ degree polynomials, to be evaluated at the $n/2$ powers of the principal $n/2$-th root of unity. Since $\omega^{2((n/2)+j)} = \omega^n \omega^{2j} = \omega^{2j}$, and $\omega^{n/2+j} = -\omega^j$, we can write the above as,

$$\begin{array}{rcl} f(\omega^j) & = & f_e(\bar{\omega}^j) + \omega^j f_o(\bar{\omega}^j) \\ f(\omega^{n/2+j}) & = & f_e(\bar{\omega}^j) - \omega^j f_o(\bar{\omega}^j) \end{array}$$

where $\bar{\omega} = \omega^2$ and for $j = 0, \ldots, n/2 - 1$.

The recursive algorithm is now evident for the case $n = 2^m$, a pure power of two. Given $a_0, \ldots, a_{n-1}$, take the FFT of $a_0, a_2, \ldots, a_{n-2}$ and $a_1, a_3, \ldots, a_{n-1}$ by (recursively) evaluating these $n/2-1$ degree polynomials,

$$f_e(y) = \sum_{j=0}^{n/2-1} a_{2j} y^j, \ f_o(y) = \sum_{j=0}^{n/2-1} a_{2j+1} y^j$$

at the powers of the $n/2$-th root of unity $\bar{\omega} = e^{\pi i/n}$,

$$b_j = f_e(\bar{\omega}^j), \ c_j = f_o(\bar{\omega}^j)$$

then combine,

$$\hat{a}_j = b_j + \omega^j c_j, \ \hat{a}_{j+n/2} = b_j - \omega^j c_j$$

where $\omega = e^{2\pi i/n}$ and $j = 0, \ldots, n/2 - 1$. The combining step takes linear time, so the recursion for run time is $T(n) = 2T(n/2) + O(n)$, so $T(n) = O(n \log n)$.

Although this recursive description is complete, implementable, and simple, application to hardware will require a non-recursive description. This is an instance of a simple recursive algorithm whose non-recursive equivalent is complicated, due to the tricky re-indexing of variables. It is organized around the *butterfly circuit*, which is a three-input, two-output circuit, $\mathcal{C}(x_0, x_1, w) = (y_0, y_1)$,

$$y_0 = x_0 + w x_1, \ y_1 = x_0 - w x_1$$

There are $logn$ layers of such butterfly circuits, one for each level of recursion, and $n/2$ circuits in each layer, to preform the combining. The tricky part is getting the right inputs to each circuit.

Example calculation

The n-th root of of unity is denoted $\omega_n = e^{2\pi i/n}$. The notation of this section gives the powers of roots o unity as the zeroth, the first, etc, $\{\,\omega^0, \omega^1, \ldots, \omega^{n-1}\,\}$, so the second roots of unity ω_2 are ordered as $\{\,1, -1\,\}$, and the fourth roots ω_4 as $\{\,1, i, -1, -i\,\}$. The conjugate roots are $\bar{\omega}_2 = \{\,-1, 1\,\}$ and $\bar{\omega}_4 = \{\,1, -i, -1, i\,\}$.

To walk through the fourier transform of the corresponding sample sets, two and four, we have,

$$a_0 + a_1 y = \begin{cases} \hat{a}_0 &= a_0 + a_1 \quad y = 1 \\ \hat{a}_1 &= a_0 - a_1 \quad y = -1 \end{cases}$$

$$a_0 + a_1 y + a_2 y^2 + a_3 y^3 = \begin{cases} \hat{a}_0 &= a_0 + a_2 + a_1 + a_3 \quad y = 1 \\ \hat{a}_1 &= a_0 - a_2 + a_1 i - a_3 i \quad y = i \\ \hat{a}_2 &= a_0 + a_2 - a_1 - a_3 \quad y = -1 \\ \hat{a}_3 &= a_0 - a_2 - a_1 i + a_3 i \quad y = -i \end{cases}$$

and the return trip,

$$\hat{a}_0 + \hat{a}_1 y = \begin{cases} \hat{a}_0 + \hat{a}_1 = a_0 + a_1 + a_0 - a_1 = 2a_0 \quad y = 1 \\ \hat{a}_0 - \hat{a}_1 = a_0 + a_1 - a_0 + a_1 = 2a_1 \quad y = -1 \end{cases}$$

$$\hat{a}_0 + \hat{a}_1 y + \hat{a}_2 y^2 + \hat{a}_3 y^3 = \begin{cases} \hat{a}_0 + \hat{a}_1 + \hat{a}_2 + \hat{a}_3 = 4a_0 \quad y = 1 \\ (\hat{a}_0 - \hat{a}_2) + i(\hat{a}_1 - \hat{a}_3) = 4a_1 \quad y = -i \\ \hat{a}_0 - \hat{a}_1 + \hat{a}_2 - \hat{a}_3 = 4a_2 \quad y = -1 \\ (\hat{a}_0 - \hat{a}_2) - i(\hat{a}_1 - \hat{a}_3) = 4a_3 \quad y = i \end{cases}$$

Graphs of overtones

The powers of the roots of unity were described as overtones of a fundamental sinewave of frequency $\omega = 2\pi/n$,

$$e^{2\pi ijk/n} = \cos k\omega j + i \sin k\omega j.$$

We illustrate this with graphs for the case $n = 6$. A connection with the *Nyquist sampling limit* is also noted.

Draw the six roots of unity in the complex plane. The principal sixth root is $\omega = (1 + \sqrt{3}i)/2$. We can make a table giving the real part (x coordinate) of ω^{jk}, that is, the k-th sample of the j-th overtone.

$j\backslash k$	0	1	2	3	4	5
0	1	1	1	1	1	1
1	1	1/2	-1/2	-1	-1/2	1/2
2	1	-1/2	-1/2	1	-1/2	-1/2
3	1	-1	1	-1	1	-1
4	1	-1/2	-1/2	1	-1/2	-1/2
5	1	1/2	-1/2	-1	-1/2	1/2

Fit a cosine wave through these points. For the zero-th overtone, we have a flat line. For the first overtone we have one period of a cosine. For the second overtone, it's perhaps not so clear, but we have samples of two periods of a cosine: three samples per period. For the third, we have samples of three periods of a cosine. This is easy, we have exactly two samples per period, at its to (1) and its bottom (-1).

After this, the samples seem to repeat. The fourth looks like the second, the fifth like the first. However, if we were to also look at the imaginary part, we would see that they are distinguishable. For instance, the first overtone the first sample is positive but for the fifth it is negative. However, it is not possible to represent a wave with frequency faster than two samples per period. After this, the waves fold back, matching waves with progressively slower frequencies. This is *aliasing* which begins at two samples per period, which is the Nyquist limit.