Primer on the Discrete Fourier Transform

January 12, 2009

1 The Discrete Fourier Transform

> *In mathematics, the discrete Fourier transform (DFT) is one of the specific forms of Fourier analysis. It transforms one function into another, which is called the frequency domain representation, or simply the DFT, of the original function (which is often a function in the time domain). But the DFT requires an input function that is discrete and whose non-zero values have a limited (finite) duration. Such inputs are often created by sampling a continuous function, like a person's voice. And unlike the discrete-time Fourier transform (DTFT), it only evaluates enough frequency components to reconstruct the finite segment that was analyzed. Its inverse transform cannot reproduce the entire time domain, unless the input happens to be periodic (forever). Therefore it is often said that the DFT is a transform for Fourier analysis of finite-domain discrete-time functions. Since the input function is a finite sequence of real or complex numbers, the DFT is ideal for processing information stored in computers. The DFT can be computed efficiently in practice using a fast Fourier transform (FFT) algorithm. Since FFT algorithms are so commonly employed to compute the DFT, the two terms are often used interchangeably in colloquial settings.* (From the Wikepedia)

Let $h[0], h[1], \cdots h[N-1]$ be a sequence of complex numbers sampled at times $0, \Delta_t, \cdots (N-1)\Delta_t$, where Δ_t is the *sampling period*. We note (see Appendix) that the sequence $h[0] \cdots h[N-1]$ can always be expressed as a weighted sum of complex sinusoids

$$h[n] = \sum_{k=0}^{N-1} a_k e^{j2\pi \frac{n}{N} k} \tag{1}$$

$$\tag{2}$$

where

$$a_k \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k}{N} n} \tag{3}$$

We now define the continuous time function $h()$ as an approximation to the original continuous time function from which the $h[]$ sequence was obtained

$$h(x\Delta_t) = \sum_{k=0}^{N-1} a_k e^{j2\pi \frac{k}{N}\, x}, \quad \text{for } x \in \mathcal{R} \tag{4}$$

or equivalently

$$h(t) = \sum_{k=0}^{N-1} a_k e^{j2\pi \frac{k}{N} \frac{t}{\Delta_t}} \tag{5}$$

$$= \sum_{k=0}^{N-1} a_k e^{j2\pi f_k\, t} \tag{6}$$

$$\tag{7}$$

where

$$f_k \stackrel{\text{def}}{=} \frac{k}{N\Delta_t} \tag{8}$$

Thus,

$$\hat{h}(f) = \sum_{k=0}^{N-1} a_k\ \delta(f - f_k) \tag{9}$$

We then define the DFT $\hat{h}[]$ in terms of the a_k coefficients that determine the FT of $h()$

$$\hat{h}[k] = N a_k = \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k}{N} n} \tag{10}$$

Note that $h[k]$ represents the amplitude of the impulse δ of the Fourier transform of $h()$ evaluate at the frequency $f_k = k/(N\Delta_t)$, which is k times the fundamental frequency $1/(k\Delta_t)$.

1.1 Periodicity

The approximating coontinous time function $h()$ is periodic with period $N\Delta_t$. To see why note

$$h(t + N\Delta_t) = \sum_{k=0}^{N-1} a_k e^{j2\pi k \frac{t+N\Delta_t}{N\Delta_t}} \tag{11}$$

$$= \sum_{k=0}^{N-1} a_k e^{j2\pi k \frac{t}{N\Delta_t}} e^{j2\pi k} = \sum_{k=0}^{N-1} a_k e^{j2\pi k \frac{t}{N\Delta_t}} = h(t) \tag{12}$$

where we used the fact that for any integer k

$$e^{j2\pi k} = \cos(2\pi k) + j \sin(-2\pi k) = 1 \tag{13}$$

The DFT is also periodic with period N, i.e.

$$h[k + N] = h[k - N] = h[k] \tag{14}$$

To see why note

$$h[k + N] = \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k+N}{N} n} \tag{15}$$

$$= \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k}{N} n} e^{-j2\pi n} \tag{16}$$

$$= \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k}{N} n} = h[k] \tag{17}$$

1.2 Interpreting of the DFT

We interpet the sequence $h[0], \cdots, h[N-1]$ as samples from the periodic continuous time signal $h()$ defined as follows

$$h(t) = \sum_{k=0}^{N-1} a_k e^{j2\pi \frac{k}{N} \frac{t}{\Delta_t}} \tag{18}$$

$$= \sum_{k=0}^{N-1} a_k e^{j2\pi f_k \, t} \tag{19}$$

$$f_k \stackrel{\text{def}}{=} \frac{k}{N\Delta_t} \tag{20}$$

$$a_k \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k}{N} n} \tag{21}$$

We note that $h()$ is one of the infinite possible set of continuous time signals that could have generated the sequence $h[]$. As such we view it as an approximation to the original signal. The approximating signal $h()$ is periodic with *fundamental frequency* $F_0 = 1/(N\Delta_t)$ and with $N-1$ consecutive harmonics of the fundamental frequency. We note the fundamental frequency, and harmonics, are measured in cycles per unit of time, where the unit of time is the same as the one used for the sampling period Δ_t. For example, if Δ_t is measured in seconds, then F_o is measuredin cycles per second (i.e., Hertz). If we choose Δ_t itself as the unit of measurement then $\Delta_t = 1$ and F_o is measured in cycles per sample.

The output of DFT is an N-dimensional vector. For N odd the vector is typically organized as follows

Index	Frequency
0	0
1	F_o
2	$2F_o$
$\cdots$	
$\frac{N-1}{2}$	$\frac{N-1}{2}F_o$
$\frac{N-1}{2}+1$	$(1-\frac{N+1}{2})F_o$
$\frac{N-1}{2}+2$	$(2-\frac{N+1}{2})F_o$
$\cdots$	
$N-2$	$-2F_o$
$N-1$	$-F_o$

To construct the table we used the fact that since the DFT is periodic

$$\frac{N-1}{2} + k \equiv \frac{N-1}{2} + k - N = k - \frac{N+1}{2} \tag{22}$$

$$N - k \equiv N - k + N = -k \tag{23}$$

If $N = 5$ the DFT would be organized as follows

Index	Frequency
0	0
1	F_o
2	$2F_o$
3	$-2F_0$
4	$-1F_0$

Note the transition from positive to negative frequencies occurs for the first positive frequency whose value would be equal to or larger than the *Nyquist frequency*

$$\frac{1}{2\Delta_t} = \frac{N}{2}F_o \tag{24}$$

If N is even the vector is organized as follows

Index	Frequency
0	0
1	F_o
2	$2F_o$
$\cdots$	
$\frac{N}{2}$	$\pm\frac{N}{2}F_o$
$\cdots$	
$N-2$	$-2F_o$
$N-1$	$-F_o$

Note the transition from positive to negative frequencies also occurs for the first positive frequency whose value would equal to or be larger than the Nyquist frequency. For example, if $N = 6$ the vector would be organized as follows

Index	Frequency
0	0
1	F_o
2	$2F_o$
3	$\pm 3F_0$
4	$-2F_o$
5	$-1F_o$

2 Appendix

2.1 The DFT

A sequence $h[0], \cdots, h[N-1]$ of complex numbers can be represented as a sum of complex sinusoids

$$h[n] = \sum_{k=0}^{N-1} a_k e^{j2\pi \frac{k}{N} n} \tag{25}$$

where

$$a_k = \frac{1}{N} \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{n}{N} k} \tag{26}$$

Informal Proof:

$$\sum_{k=0}^{N-1} a_k e^{j2\pi \frac{k}{N} n} = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{n}{N} k} e^{j2\pi \frac{k}{N} u} \tag{27}$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{n=0}^{N-1} h[n] e^{-j2\pi k \frac{u-n}{N}} = \frac{1}{N} \sum_{k=0}^{N-1} h[u] + \sum_{n \neq u} h[n] \sum_{k=0}^{N} e^{-j2\pi \frac{u-n}{N} k} \tag{28}$$

$$h[u] + \sum_{n \neq u} h[n] \sum_{k=0}^{N} e^{-j2\pi \frac{k}{N}} = h[u] \tag{29}$$

For the last two steps we used the fact that $e^{j2\pi} = e^{j2\pi(u-n)}$. Moreover note that the complex numbers $e^{j2\pi 0} = e^{j2\pi 1/N} \cdots = e^{j2\pi (N-1)/N}$ are vectors of lenght 1, equidistantly located around the complex unit circle. Therefore their sum is at the center of the complex unit circle, i.e.,

$$\sum_{k=0}^{N} e^{-j2\pi \frac{k}{N}} = 0 \tag{30}$$

2.2 Using the Fast Fourier Transform in Matlab

```
% Illustrates how to get the discrete time FFT of a Gaussian function and
% how to interpret it with respect to the continuous time Fourier Transform
```

```
%
% Copyright @ Javier R. Movellan, UCSD, 2009

clear
clf

% (1) Choose  a unit of measurement for the time axis, e.g. seconds.
% (2) Choose the sampling period using the chosen units
dt= 0.1; % time in seconds.If dt =1; the unit of measurement is the time
         % between samples and the frequency is measured in cycles per
         % sample.
% (3)  Choose the number of samples
N =35;

% (4) The fft assumes the samples  come from a periodic signal starting at
% t=0, dt, 2 dt, ... (N-1) dt. The period of the signal is
% nSamples*dt. Thus implicitly we are defining how the signal would
% behave for negative values of t. We have that s(t  - N dt) = s(t). Thus
% s((N-k) dt ) = s( (N-k -N ) dt ) = s(-kdt)

for n=1:N
  if (n-1< N/2)
    t(n) = (n-1)* dt; % we start at time 0
  else
    t(n) = ((n-1) - N)*dt;
  end
end

% (5) We now populate our vector with the signal we want to get the fft
% for.

a= 3;
s = exp(-pi*(a*t).^2);

% (6) Get the fft

S = fft(s);
```

```
% (6) The k element of the fft vector is the continuous fourier transform of the
% signal evaluated at the frequency f = (k-1)* 1/(N dt), so S[1] = F(0), S[2]
% = F(1/dt), S(3) = F(2/dt).  However the fft is also periodic so F( f) =
% F(f+ 1/dt) = F(1 - 1/dt). Thus S[N-k] = F((N-k+N)/(N dt)) = F (-k /(Ndt))
% . Below we rearrange things so the negative frequencies are to the left
% and the positive frequencies to the right

for n=1: N
  if(n-1 < N/2)
    f(n) = (n-1)/(N*dt); % frequency in cycles per unit of time
  else
    f(n) = ((n-1)-N)/(N*dt); % frequency in cycles per unit of time
  end
end

% Plot and verify the results

subplot(6,1,1)
scatter(t,s)
xlabel('Time in Seconds')
% To verify things we'll plot the half magnitude interval
line(0.4697 *[1/a 1/a] , [0 max(s)])
line(-0.4697 *[1/a 1/a] , [0 max(s)])
line([min(t), max(t)]  ,max(s)* [0.5 0.5])

subplot(6,1,2)
scatter(f,abs(S))
xlabel('Frequency in Hertz')
ylabel('Magnitude')
% To verify things we'll plot the half magnitude interval
line(0.4697 *[a a] , [0 max(S)])
line(-0.4697 *[a a] , [0 max(S)])
line([min(f), max(f)]  ,max(S)* [0.5 0.5])



% (7) If real(S) or imag(S) is very small the phase spectrum runs into
% numerical issues with matlab. Just make very small numbers exactly zero
```

```
tiny= 10^(-10);
for n=1:N
  if(abs(real(S(n)) < tiny)) S(n) = j*imag(S(n)); end
  if(abs(imag(S(n)) < tiny)) S(n) = real(S(n)); end
end

subplot(6,1,3)
scatter(f,phase(S))
xlabel('Frequency in Hertz')
ylabel('Phase')




% (8) The matlab functions fftshift and ifftshift  can be used to
% simplify all the index shifting.  Here is how to use them

subplot(6,1,4)

t2 = dt*(- floor(N/2):floor(N/2));
s2 = exp(-pi*(a*t2).^2);
s2= ifftshift(s2);
% we Note that g2 =s
xlabel('Double Check')
S2 = fft(s2);
S2= ifftshift(S2);
% verify we get same results
subplot(6,1,4)
scatter(1:N, fftshift(s2))
xlabel('Double Check')
subplot(6,1,5)
scatter(1:N, abs(S2))
xlabel('Double Check')
subplot(6,1,6)
for n=1:N
  if(abs(real(S2(n)) < tiny)) S2(n) = j*imag(S2(n)); end
  if(abs(imag(S2(n)) < tiny)) S2(n) = real(S2(n)); end
```

```
end
scatter(1:N, phase(S2))
xlabel('Double Check')
```

2.3 Using the Matlab conv function

```
% Filters a signal using the conv function
%
% s : The signal
% k : The filter kernel (time flipped impulse response
% c:  Integer with index for origin  of he kernel, points to
%     the right of the origin are non-causal.
% z:  The output of the filter to each point of s. The outputs for the
%     left and right limits of s are obtained by zero padding s.
%
%
% Example1; Filter a signal s using a causal ramp filter with 4 time steps
% z = doFilter(s, 1:4, 4)
%
% Example2; Filter a signal s using a ramp filter that looks 3 steps
% into the future
% z = doFilter(s, 1:4, 1)
%
% Copyright Javier R. Movellan,2009, UCSD
function z = doFilter(s,k,c)


z=conv(s,fliplr(k)); % note we need to convolve with the impulse response,
                     % which is the time flipped version of the kernel


%Matlab automatically pads s with zeros to the left and right.
% We now get rid of the response of the filter at times other than the
% times of the original data

z(1:(length(k)-c))=[];
z(end-(c-2):end)=[];
```