Digital Signal Processing For Audio Applications

Anton Kamenov
RecordingBlogs.com
Foreword

In the summer of 2003 we began designing multi-track recording and mixing software – Orinj at RecordingBlogs.com – a software application that will take digitally recorded audio tracks and will mix them into a complete song with all the needed audio production effects. Manipulating digital sound, as it turned out, was not easy. We had to find the answers of many questions, including what digital audio was, how we could mix audio tracks, how we could track the amplitude of digital sound so that we could apply compression, how we could track frequencies so that we could equalize, what a good model of artificial reverb would be, and many others. Bits of relevant information were available, albeit not always well organized and not always intuitive.

"Digital Signal Processing for Audio Applications" provides much of the needed information. It is a simple structured approach to understanding how digitally recorded sound can be manipulated. It presents and explains, and sometimes derives, the mathematical theory that the DSP user can employ in designing sound manipulating applications.

Although this book introduces much mathematics, we have designed it not for mathematicians, but for the engineers and hobbyists, who would be interested in the practical applications of DSP and not in its theoretical derivations. If properly explained, much of the practical DSP applications reduce to simple algebra. This said, we have included a sufficient amount of theory to provide an explanation of why DSP works the way it does. It is important for practitioners to have a good understanding of how DSP concepts come about. Much of the available DSP information has too much theory and not enough examples. Much of it has too many practical examples and not enough theoretical backing. We hope to have found the proper balance.

We hope you enjoy this book and make use of its definitions, explanations, and numerous examples.

The author and the administrators of www.recordingblogs.com

Version 1.0.0
July 2012
Licensed to: John Smith

© 2011 Anton Kamenov. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the prior written permission of the author.
Chapter 12. The continuous and discrete Fourier transforms

Approximating a signal with the Fourier series results in a collection of simple cosine and sine waves with varying frequency and amplitude. Alternatively, with the use of trigonometric identities as in equation 5.2, the Fourier series can be written as a collection of simple waves with varying frequency, amplitude, and phase. Once frequencies, amplitudes, and phase amounts are known, the value of the signal at a point in time is known and the signal is a function of time. Alternatively, once amplitudes and phase amounts are known, the signal at a specific point in time becomes a function of the frequency. Simply put, if we know amplitudes, frequencies, and phase amounts, then all we need is time. If we know time and amplitudes, all we need is frequencies and phase amounts.

12.1. Definitions

The Fourier transform translates something that is a function of time given specific amplitudes, frequencies, and phase amounts into something that is a function of frequencies and phase amounts for specific amplitudes and time. As any other transform, the Fourier transform translates one type of function into another. We cannot derive the Fourier transform as it is not something that is derived. It is simply something that is defined.

The continuous Fourier transform of the function \( x(t) \) is defined as follows.

\[
H(s) = \int_{-\infty}^{\infty} x(t)e^{-2\pi j t s} dt \tag{12.1}
\]

The inverse continuous Fourier transform is

\[
x(t) = \int_{-\infty}^{\infty} H(s)e^{2\pi j t s} ds \tag{12.2}
\]

The discrete Fourier transform (DFT) of the function \( x(k) \) is defined as follows.

\[
H(n) = \sum_{k=0}^{N-1} x(k)e^{-\frac{2\pi j k n}{N}}, \quad n = 0, 1, ..., N - 1 \tag{12.3}
\]

The inverse discrete Fourier transform (inverse DFT) is

\[
x(k) = \frac{1}{N} \sum_{n=0}^{N-1} H(n)e^{\frac{2\pi j k n}{N}}, \quad k = 0, 1, ..., N - 1 \tag{12.4}
\]

Euler’s formula\(^{10}\) allows us to write, for example, the discrete Fourier transform also as follows.

\(^{10}\) Euler’s formula states that \( e^{ix} = \cos(x) + j \sin(x) \).
Chapter 12. The continuous and discrete Fourier transforms

\[
H(n) = \sum_{k=0}^{N-1} x(k) \left( \cos \left( \frac{2\pi k n}{N} \right) - j \sin \left( \frac{2\pi k n}{N} \right) \right) \tag{12.5}
\]

\[
x(k) = \frac{1}{N} \sum_{n=0}^{N-1} H(n) \left( \cos \left( \frac{2\pi k n}{N} \right) + j \sin \left( \frac{2\pi k n}{N} \right) \right) \tag{12.6}
\]

We have already used formulae similar to the ones above. The frequency content computation of Chapter 5 is simply the forward Fourier transform. In Chapter 7 we derived the general form of a low pass filter by taking a desired magnitude response, which is a function of the frequency, summing up all cosine functions that fit in that desired magnitude response, and scaling them by \((2 / f_s)\). The filter \(a(k)\) was computed as the sum of integer frequencies \(m\) up to \(f_c\), which is simply a form of the inverse Fourier transform. We will confirm these findings after a simple DFT example.

12.2. Simple DFT example

Suppose that we take a simple sine wave and sample it over \(N\) points for a period of 1 second and suppose that the simple wave has the integer frequency \(f\), amplitude \(A\), and phase \(m\). The values of this simple wave would be

\[
A \sin \left( \frac{2\pi f (k - m)}{N} \right)
\]

\[
= A \cos \left( \frac{2\pi f k}{N} \right) \sin \left( \frac{2\pi f m}{N} \right) - A \sin \left( \frac{2\pi f m}{N} \right) \cos \left( \frac{2\pi f k}{N} \right) = B \cos \left( \frac{2\pi f k}{N} \right) + C \sin \left( \frac{2\pi f k}{N} \right) \tag{12.7}
\]

where \(B = -A \sin(2\pi f m / N)\) and \(C = A \cos(2\pi f m / N)\). We can compute the DFT as follows.

\[
H(n) = H \left( \sin \left( \frac{2\pi f k}{N} \right) \right)
\]

\[
= \sum_{k=0}^{N-1} \left( B \cos \left( \frac{2\pi f k}{N} \right) + C \sin \left( \frac{2\pi f k}{N} \right) \right) \left( \cos \left( \frac{2\pi n k}{N} \right) - j \sin \left( \frac{2\pi n k}{N} \right) \right) \tag{12.8}
\]

for \(n = 0, 1, ..., N - 1\). Here we note that most of these functions will be orthogonal over the sampling period, except in two cases – for \(n = f\) and for \(N - n = f\). In the second case, we note that
Chapter 12. The continuous and discrete Fourier transforms

\[
\cos \left( \frac{2\pi k (N - n)}{N} \right) = \cos \left( 2\pi k - \frac{2\pi k n}{N} \right) = \cos \left( \frac{2\pi k n}{N} \right) \quad (12.9)
\]

\[
\sin \left( \frac{2\pi k (N - n)}{N} \right) = \sin \left( 2\pi k - \frac{2\pi k n}{N} \right) = -\sin \left( \frac{2\pi k n}{N} \right)
\]

The transform, as a function of \( f \), translates to

\[
H(n) \approx \begin{cases} 
B \sum_{k=0}^{N-1} \cos^2 \left( \frac{2\pi k n}{N} \right) - j C \sum_{k=0}^{N-1} \sin^2 \left( \frac{2\pi k n}{N} \right), & n = f, N - f \\
0, & n \neq f, N - f
\end{cases} \quad (12.10)
\]

From the computations of RMS amplitude in Chapter 4 we know that the two sums for a large enough \( N \) evaluate to \( N / 2 \) and so the transform is simply

\[
H(n) \approx \begin{cases} 
\frac{N}{2} (B - j C), & n = f, N - f \\
0, & n \neq f, N - f
\end{cases} \quad (12.11)
\]

Thus, the DFT of a simple wave with frequency \( f \) produces a function \( H(n) \) that indicates something about the single wave at components \( n = f \) and \( n = N - f \) and is zero otherwise. Specifically, at \( n = f \) and \( n = N - f \)

\[
|H(n)| \approx \sqrt{\text{Re}(H(n))^2 + \text{Im}(H(n))^2} = \frac{N}{2} \sqrt{B^2 + C^2} = \frac{N}{2} A \quad (12.12)
\]

Similarly

\[
\text{atan2}(\text{Im}(H(n)), \text{Re}(H(n))) = \text{atan2}(N C, N B) = -\frac{2\pi f m}{N} \quad (12.13)
\]

The discrete Fourier transform \( H(n) \) of a real valued signal \( x(k) \) shows the amplitude of the \( n^{th} \) simple wave component in the real valued signal \( x(k) \)

\[
A_n = 2 \frac{|H(n)|}{N} = 2 \frac{\sqrt{\text{Re}(H(n))^2 + \text{Im}(H(n))^2}}{N} \quad (12.14)
\]

and the phase of the \( n^{th} \) simple wave component in \( x(k) \) is

\[
\phi_n = \text{atan2}(\text{Im}(H(n)), \text{Re}(H(n))) \quad (12.15)
\]

This result is no different than the result in equations 5.15 through 5.18, which we introduced during our discussion of the Fourier series. The Fourier transform is a natural extension of the Fourier series.
There are three motivations for using the Fourier transform rather than the Fourier series. First, the Fourier transform allows for shorter notation. Second, the use of complex numbers may come as a surprise, but it allows the Fourier transform to return a complex number, which means two separate values that can be used to describe the amplitude and the phase of the signal separately. Third, the Fourier transform has an inverse transform, which is easy to describe and which has its own uses and we will see below.

Suppose that we compute, for example, the DFT of a simple sine wave at 10 Hz, using 100 DFT points over 2 seconds of the wave. The magnitude of the resulting DFT components will be as in figure 52. The DFT will be non-zero – approximately – at components 20 and 80. Since we are sampling with 100 points over 2 seconds, we have effectively sampled the wave with the sampling frequency 50 Hz. This means that we have used the DFT with components at 0.5 Hz, 1 Hz, 1.5 Hz, …, 49.5 Hz. The 20th component is 10 Hz and it makes sense that we have a non-zero value at this component. According to the discussion above, we also get a non-zero value at component 80 = 100 – 20. The magnitude of the DFT at these components is 100 / 2 = 50. These results are as expected.

When the input to the DFT or the inverse DFT is real valued and not complex, the DFT produces complex numbers such that \( H(N - 1 - n) \) is the complex conjugate of \( H(n) \). The information for \( n > N / 2 \) is redundant.

The fact that the DFT of a real valued signal produces repeated and redundant information is uncomfortable. We have two choices. First, we can ignore the second half DFT. Second, we can feed the DFT with complex numbers. The complex frequency 10 Hz sampled over \( N \), for example, is

\[
e^{-\frac{2\pi}{N} 10 k}
\]

When the DFT is used on complex data, however, the results are different. Take the simple wave

\[x(k) = A_0 + A_1 \sin(2\pi k f / N)\]

\[A_1 \approx 50\]
Chapter 12. The continuous and discrete Fourier transforms

\[ e^{-\frac{2\pi j f k}{N}} = \cos \left( \frac{2\pi f k}{N} \right) - j \sin \left( \frac{2\pi f k}{N} \right) \]  \hspace{1cm} (12.17)

The component \( H(n) \) of the DFT at \( n = f \) would be

\[
H(n) \approx \sum_{k=0}^{N-1} \left( \cos^2 \left( \frac{2\pi k n}{N} \right) + \sin^2 \left( \frac{2\pi k n}{N} \right) \right) - j \sum_{k=0}^{N-1} \left( \sin \left( \frac{2\pi k n}{N} \right) \cos \left( \frac{2\pi k n}{N} \right) + \sin \left( \frac{2\pi k n}{N} \right) \cos \left( \frac{2\pi k n}{N} \right) \right) \]  \hspace{1cm} (12.18)

\[
= N
\]

The difference is that this result is twice as large, as the result of the DFT of a real valued signal.

The discrete Fourier transform \( H(k) \) of a complex valued signal \( x(k) \) shows the amplitude of the \( n^{th} \) simple wave component in the complex valued signal \( x(k) \)

\[
A_n = \frac{|H(n)|}{N} = \sqrt{\text{Re}(H(n))^2 + \text{Im}(H(n))^2} \]  \hspace{1cm} (12.19)

and the phase of the \( k^{th} \) simple wave component in \( x(k) \) is

\[
\Phi_n = \arg(H(n)) = \text{atan}2(\text{Im}(H(n)), \text{Re}(H(n))) \]  \hspace{1cm} (12.20)

12.3. The DFT and FIR filters

Since the low pass filter of Chapter 7 is simply a complex signal composes of the simple waves with frequency less than the cutoff frequency \( f_c \), we should be able to use the DFT to compute the frequency (magnitude and phase) response of the low pass filter. As above, we note that this low pass filter is real valued and hence we can dispose of half of the DFT components. We also note that we have already scaled the low pass filter of length \( N \) by \( 2 / N \). Thus, we can compute the standard DFT over \( N \) points, but in order to compute the magnitude, we will not scale by \( 2 / N \). The resulting computation for the magnitude response of the filter is equivalent to the one we already used in equation 8.5.

\[
|H(n)| = \sqrt{\left( \sum_{k=0}^{N-1} a(k) \sin \left( \frac{2\pi k n}{N} \right) \right)^2 + \left( \sum_{k=0}^{N-1} a(k) \cos \left( \frac{2\pi k n}{N} \right) \right)^2} \]  \hspace{1cm} (12.21)
Chapter 12. The continuous and discrete Fourier transforms

The fact that the DFT is computed with a small set of components – the length of the filter – means that the magnitude response would be sparse in the frequency domain, as shown on figure 53. The magnitude response in figure 53 employs a DFT of \( N = 101 \) points, which is the same as the length of the filter. The cutoff frequency is between components 2 and 3, which, given the sampling frequency of \( f_s = 2000 \) Hz, translates to between 39.6 Hz and 59.4 Hz (sampling 101 times over 2000 Hz implies 19.8 Hz per component). Note that figure 53 is limited to component 50, or half of \( N \).

With the DFT we can also compute the phase response of our low pass filter prototype. We note that the filter has coefficients that are symmetric around the middle, i.e., \( a(k) = a(N - 1 - k) \). Assume for a moment, that \( \text{Re}(H(n)) > 0 \). Then

\[
\phi_n = \text{atan2}(\text{Im}(H(n)), \text{Re}(H(n))) = \text{atan}\left(\frac{\text{Im}(H(n))}{\text{Re}(H(n))}\right) = \text{atan}\left(\frac{\sum_{k=0}^{N-1} a(k) \sin\left(\frac{2\pi k n}{N}\right)}{\sum_{k=0}^{N-1} a(k) \cos\left(\frac{2\pi k n}{N}\right)}\right) \tag{12.22}
\]

Since \( a(k) = a(N - 1 - k) \),

\[
a(k) \sin\left(\frac{2\pi k n}{N}\right) + a(N - 1 - k) \sin\left(\frac{2\pi (N - 1 - k) n}{N}\right)
= 2 a(k) \sin\left(\frac{2\pi (N - 1)n}{2 N}\right) \cos\left(\frac{2\pi (N - 1 - 2 k)n}{2 N}\right) \tag{12.23}
\]

Similarly,

\[
a(k) \cos\left(\frac{2\pi k n}{N}\right) + a(N - 1 - k) \cos\left(\frac{2\pi (N - 1 - k) n}{N}\right)
= 2 a(k) \cos\left(\frac{2\pi (N - 1)n}{2 N}\right) \cos\left(\frac{2\pi (N - 1 - 2 k)n}{2 N}\right) \tag{12.24}
\]

The phase response, assuming that the filter is of odd length, is as follows.
\[ \Phi_n = \text{atan} \left( \frac{2 \sin \left( \frac{2 \pi n N - 1}{N} \right)}{2 \cos \left( \frac{2 \pi n N - 1}{N} \right)} \left( 1 + \sum_{k=0}^{(N-1)/2} \left( a(k) \cos \left( \frac{2 \pi n N - 1 - 2k}{2} \right) \right) \right) \right) \]  
(12.25)

The "1 +" in the numerator and denominator handle the case when the filter is of odd length \( N \). If the filter has even number of points \( N \), the computations would be the same but without "1 +". Thus,

\[ \Phi_n = \text{atan} \left( \frac{\sin \left( \frac{2 \pi n N - 1}{N} \right)}{\cos \left( \frac{2 \pi n N - 1}{N} \right)} \right) = \frac{2 \pi n N - 1}{N} \]  
(12.26)

**Fact:** The linear phase response of symmetric FIR filters is used in radars. Radars determine distance by measuring the phase of a number of frequencies in the frequency spectrum. If the radar should use a FIR filter, it is helpful if the FIR filter introduces minimal or at least well known distortion of the phase.

As discussed in Chapter 8, the phase response is linear with respect to the frequency \( n \) and produces a delay of \((N - 1) / 2\) samples. These computations assume that \( \text{Re}(H(n)) > 0 \), but are similar in other cases.

### 12.4. Using the inverse DFT

If the DFT produces the magnitude response of a filter, then we should be able to produce a filter by taking the inverse DFT from a desired magnitude response. In Chapter 5 we created a low pass filter by summing up the simple waves up to the cutoff frequency.

\[ a(k) = \frac{2}{f_s} \sum_{m=1}^{f_c} \cos \left( \frac{2 \pi m k}{f_s} \right), \quad k = 0, 1, ..., f_s - 1 \]  
(12.27)

This equation can also be written as

\[ a(k) = 2 \left( \frac{1}{f_s} \sum_{m=1}^{f_c} B_m \cos \left( \frac{2 \pi m k}{f_s} \right) \right), \quad B_m = \begin{cases} 1, & m \leq f_c \\ 0, & m > f_c \end{cases} \]  
(12.28)

where \( B_m \) is the desired magnitude response of the filter, equal to 1 up to the cutoff frequency \( f_c \) and zero afterwards. The upper limit of the sum is \( f_s / 2 \) as suggested by the Nyquist-Shannon sampling theorem.

This formula is very similar to the inverse DFT, except that it only takes the real part of the inverse DFT, hence only the cosines, is limited to half of the inverse DFT, hence the sum goes up to only \( f_c / 2 \), and is scaled by 2. Thus, this formula samples the frequency content and uses computations similar to the inverse DFT to create a filter. We are not limited to sampling only at \( f_c \), and we can sample at any evenly spaced \( N \) points. In other words, the low pass filter can be created by
Chapter 12. The continuous and discrete Fourier transforms

\[
a(k) = 2 \left( \frac{1}{N} \sum_{n=0}^{N/2} B_n \cos \left( \frac{2\pi n k}{N} \right) \right), \quad B_n = \begin{cases} 
1, & \frac{n f_s}{N} \leq f_c \\
0, & \frac{n f_s}{N} > f_c 
\end{cases}, \quad k = 0, 1, N - 1 \quad (12.29)
\]

We note that cosine functions are symmetric around the origin and that the filter \(a(k)\) could similarly be symmetric around the origin, if we are after a linear phase response. We can then extend the sum and remove the scaling factor of 2, for \(k = 0, 1, ..., N - 1\) as follows.

\[
a(k) = \frac{1}{N} \sum_{n=-N/2}^{N/2} B_n \cos \left( \frac{2\pi n k}{N} \right), \quad B_n = \begin{cases} 
1, & \frac{n f_s}{N} \leq f_c \\
0, & \frac{n f_s}{N} > f_c 
\end{cases} \quad (12.30)
\]

Note the natural addition of \(n = 0\) to the sum. Since \(B_n\) represents the desired filter magnitude response, the magnitude response should remain at 1 as the frequency approaches zero.

Finally, in an attempt to make the filter symmetric around \((N - 1) / 2\), for the same \(B_n\) and \(k\), we shift the filter as follows.

\[
a(k) = \frac{1}{N} \sum_{n=-N/2}^{N/2} B_n \cos \left( \frac{2\pi n \left( k - \frac{N - 1}{2} \right)}{N} \right) \quad (12.31)
\]

This filter formula resembles the inverse DFT, but it is not the inverse DFT. We do not use the inverse DFT to produce a FIR filter\(^{13}\). There are important differences between the inverse DFT and the filter computation above. First, the formula above uses only the real part of the inverse DFT. Second, the formula is shifted both in \(n\), as the sum runs from \(-N / 2\) to \(N / 2\) and in \(k\), as the sum uses \(k - (N - 1) / 2\).

The formula above is the real portion of a generalized or shifted form of the inverse complex valued DFT, which is usually written as

\[
X(k) = \frac{1}{N} \sum_{n=0}^{N-1} H(n) e^{2\pi j (k+a) (n+b) / N} \quad (12.32)
\]

Note that we could easily add the imaginary part of the inverse DFT to equation 12.31 above. Since sine functions are odd (i.e., \(\sin(-x) = -\sin(x)\)), the sum of the sine values for \(n = -N / 2, ..., N / 2\) would simply be zero. In other words, by designing a desired magnitude response \(B_n\) that is symmetric around the origin, we created an inverse DFT and thus a filter that contains only real valued coefficients.

Using a desired magnitude response that includes negative frequencies may be troublesome, since negative frequencies do not mean much. The technique of using a symmetric magnitude response to produce a real valued filter is, however, common.

\(^{13}\) If we wanted to use the non-generalized form of the DFT as defined in equation 12.3, we would have to scale the results by 2. If we include a value for the magnitude response at frequency zero, that value would be 0.5 and not 1.
Suppose that, as before, we need a filter is of length \( N = 101 \) and cutoff frequency \( f_c = 40 \) Hz, given the sampling frequency \( f_s = 2000 \) Hz. To use the inverse DFT to create the filter, we must first produce the desired magnitude response \(|H(n)|\). We can use the inverse DFT with \( N = 101 \) points to produce the 101 points of the filter, but we must be careful to define a magnitude response that is symmetric around the origin. We will use \( N = 101 \) DFT components between \(-f_s/2 \) and \( f_s/2 \). Each component will be responsible for \( 2000 / 101 \approx 19.8 \) Hz. DFT component 0 is at frequency 0. DFT components -1 and 1 are at -19.8 Hz and 19.8 Hz respectively. DFT components -2 and 2 are at -39.6 Hz and 39.6 Hz respectively, and so on.

An appropriate — randomly chosen — magnitude response might be \(|H(n)|\) will have values of 1 at components -2, -1, 0, 1, 2 and would be zero everywhere else. One of the benefits of using the inverse DFT, however, is that we can include a transition band in the desired magnitude response. If we want to include a transition band, we might define a magnitude response that is 1 at components -1, 0, and 1, that is 0.5 at components -2 and 2, and that is zero everywhere else. This could be an appropriate magnitude response of a low pass filter with cutoff frequency \( f_c = 40 \) Hz, as we might want a smoother transition band to mitigate the Gibbs phenomenon ripples.

The results of the inverse DFT on \(|H(n)|\) are shown on figure 54 (solid red and dashed red). The magnitude response taken with the forward DFT on figure 55 looks great, but it is misleading. Since the filter has a low cutoff frequency and since the DFT has few points, the DFT produces approximately the desired magnitude response. The more detailed magnitude response computation on figure 56 compares the response of this filter to the low pass filter created in Chapter 7. Including a transition band in the desired magnitude response \(|H(n)|\) produces a filter with a slower transition band, but with smaller ripples and better stop band attenuation.
12.5. Sampling of the continuous inverse Fourier transform

The continuous-time Fourier transform translates a signal \( x(t) \) that is a function of time for specific amplitudes, frequencies, and phase amounts into a signal \( H(x(t)) \) that is a function of frequencies and phase amounts for specific amplitudes and time. The inverse Fourier transform translates functions of frequencies and phase amounts into signals.

Suppose that rather than a discrete desired magnitude response function, we use a continuous magnitude response function.

\[
|H(f)| = \begin{cases} 
1, & -\frac{f_c}{f_s} \leq f \leq \frac{f_c}{f_s} \\
0, & |f| > \frac{f_c}{f_s} 
\end{cases}
\]  

(12.33)

This is an appropriate magnitude response, symmetric around the origin, of some low pass filter with normalized cutoff frequency \( F = f_c / f_s \), \( F < 1/2 \). The inverse Fourier transform of this function is

\[
x(t) = \int_{-F}^{F} e^{2\pi j ft} df = \frac{1}{2\pi j t} \left( e^{2\pi j ft} - e^{-2\pi j ft} \right) = \frac{1}{\pi t} \sin(2\pi F t) 
\]

(12.34)

**Fact:** The function \( \sin(t) / t \) is the function \( \text{sinc}(t) \). Thus, filters are occasionally called **sinc functions**.

Many desired magnitude responses can be converted into a filter with the inverse Fourier transform or the inverse discrete Fourier transform to produce, for example, nice parametric equalizers – equalizers in which the user can adjust both the gain and the frequencies at which changes in the gain occur.

This function has ever increasing oscillations until \( t = 0 \), at which point it is \( x(0) = 2 F \), after which it has ever decreasing oscillations. To create a discrete time filter, we sample this continuous time function between -1 and 1 at the sampling frequency. We pick a discrete filter of length \( N \) from the middle \( N \) points of the sampled \( x(t) \) and shift it to the right by \( (N - 1) / 2 \). This would produce the low pass filter below, which we described in Chapter 7 as the alternative form to our original low pass filter formula. This is, in fact, the more widely used formula and the more precise one.

**The following is the formula for a finite impulse response low pass filter of length \( N \), cutoff frequency \( f_c \) given the sampling frequency \( f_s \) and \( k = 0, 1, ..., N - 1 \). The usual transformations produce high pass, band pass, and band stop filters.**

\[
a(k) = \begin{cases} 
\sin \left( \frac{2\pi f_c \left( k - \frac{N - 1}{2} \right)}{f_s} \right), & k \neq \frac{N - 1}{2} \\
\frac{f_c}{f_s}, & k = \frac{N - 1}{2} 
\end{cases}
\]  

(12.35)
A high pass filter can be produced by subtracting the coefficient of the low pass filter from those of an all pass filter.

\[
a(k) = \begin{cases} 
\sin \left( \frac{2\pi f_c \left( k - \frac{N - 1}{2} \right)}{f_s} \right), & k \neq \frac{N - 1}{2} \\
\pi \left( k - \frac{N - 1}{2} \right), & k = \frac{N - 1}{2} 
\end{cases} \tag{12.36}
\]

A band pass filter is the difference between two low pass filters at cutoff frequencies \( f_c \) and \( f_c', f_c' < f_c \).

\[
a(k) = \begin{cases} 
\sin \left( \frac{2\pi f_c \left( k - \frac{N - 1}{2} \right)}{f_s} \right) - \sin \left( \frac{2\pi f_c' \left( k - \frac{N - 1}{2} \right)}{f_s} \right), & k \neq \frac{N - 1}{2} \\
\frac{2 f_c}{f_s} - \frac{2 f_c'}{f_s}, & k = \frac{N - 1}{2} 
\end{cases} \tag{12.37}
\]

A band stop filter could be the difference between an all pass filter and a band pass filter. If \( f_c' < f_c \)

\[
a(k) = \begin{cases} 
\sin \left( \frac{2\pi f_c \left( k - \frac{N - 1}{2} \right)}{f_s} \right) + \sin \left( \frac{2\pi f_c' \left( k - \frac{N - 1}{2} \right)}{f_s} \right), & k \neq \frac{N - 1}{2} \\
1 - 2 \frac{f_c}{f_s} + 2 \frac{f_c'}{f_s}, & k = \frac{N - 1}{2} 
\end{cases} \tag{12.38}
\]

**12.6. Deriving the family of Hamming windows**

The usual purpose of windows is to mitigate the ripples of the Gibbs phenomenon, which are the result of the Fourier series approximation – a series of continuous functions – of the discontinuous desired magnitude response. One obvious thing to do to reduce these ripples is to use a different desired magnitude response, one in which this discontinuity is not as blatant. Take a look at figure 57. Rather than using the inverse Fourier transform on the ideal magnitude response (blue), we will do so on a revised magnitude response (red).
We can recognize that the revised magnitude response is the sum of the scaled original magnitude response $H(f)$ with two scaled and shifted versions of itself.

$$\alpha H(f) + \beta H(f - f_0) + \beta H(f + f_0) \quad (12.39)$$

where $\alpha$ is the scaling factor for the original magnitude response, $\beta$ is the scaling factor for the shifted magnitude responses, and $f_0$ and $-f_0$ are the two shifts. In the example above, $\alpha = 0.5$, $\beta = 0.25$, and $f_0 = \frac{\pi}{4} f_c$, where $f_c$ is the cutoff frequency.

We can also note that the windowed filter is the product of the filter and the window. If we are to use the inverse Fourier transform on some new (presumably, windowed) magnitude response, then this desired magnitude response must be a convolution of the Fourier transform of the filter and the Fourier transform of the window. We rewrite the revised magnitude response above as a convolution.

$$H(f) * (\alpha \delta(f) + \beta \delta(f - f_0) + \beta \delta(f + f_0)) \quad (12.40)$$

where $\delta$ is the Dirac delta function. We use the Dirac delta function, as it allows us to both write the desired magnitude response as a convolution, as well as to recognize the transform of the cosine function in the last two terms. The inverse Fourier transform of this desired magnitude response is the filter before the window, multiplied by the window

$$w(k) = \alpha + 2 \beta \cos(f_0 k) \quad (12.41)$$

This is the generalized Hamming window family, which includes the Hamming window itself and the Hann window.

### 12.7. A Dolph-Chebychev window

We will put the inverse Fourier transform to more practical use with the Dolph-Chebychev window. Dolph-Chebychev windows are interesting, because they control the ripples in the pass and stop bands.

To cut down on notation, we introduce angular frequencies. Angular frequencies, usually denoted with $\omega$, measure the number of rotations per unit of time. They are typically computed in radians per second, where radians measure the ratio of an arc of a circle to the radius of a circle. Since the arc of one rotation is the full circle, or $2\pi$ radians, a frequency of one rotation per second is the frequency of $2\pi$ radians per second. Thus, we can also say that if something has the frequency $f$, it also has the angular frequency $\omega = 2\pi f$. For example, $f = 40$ Hz means $\omega = 251.3$ radians per second.

\[ \text{The frequency } f \text{ is also the angular frequency} \]

$$\omega = 2\pi f \quad (12.42)$$

$f$ is normalized, when divided by the sampling frequency $f_s$. By the Nyquist-Shannon sampling theorem, normalized frequencies are between 0 and $\frac{1}{2}$, or between $\frac{-1}{2}$ and $\frac{1}{2}$, if negative frequencies are allowed. Thus, normalized angular frequencies span the interval between $-\pi$ and $\pi$. 
When angular frequencies are normalized, the sampling frequency itself normalizes to \( \omega_s = 2\pi \). Suppose that we have the angular frequency \( \omega = 1 \) and the signal is sampled at \( f_s = 2000 \) Hz. Then in reality the frequency \( \omega \) is equivalent to \( 1 \times 2000 / 2\pi = 318.3 \) Hz.

To design the Dolph-Chebychev window, we begin with the Dolph-Chebychev desired magnitude response.

\[
|H(\omega)| = \frac{T_{2M} \left( \frac{\cos \left( \frac{\omega}{2} \right)}{\cos \left( \frac{\omega_c}{2} \right)} \right)}{T_{2M} \left( \frac{1}{\cos \left( \frac{\omega_c}{2} \right)} \right)}
\]  

(12.43)

Here \( T_n \) are the Chebychev polynomials

\[
T_n(x) = \begin{cases} 
\cos(n \text{acos}(x)), & |x| \leq 1 \\
\cosh(n \text{acosh}(x)), & |x| > 1 
\end{cases}
\]  

(12.44)

\( M \) is an integer parameter, typically set to \( M = (N - 1) / 2 \), where \( N \) is the length of the desired filter, and \( \cosh \) and \( \text{acosh} \) are the hyperbolic cosine and arccosine

\[
\cosh(x) = \frac{1}{2} \left( e^x + e^{-x} \right)
\]  

(12.45)

\[
\text{acosh}(x) = \ln(x + \sqrt{x^2 - 1}), \quad x \geq 1
\]  

(12.46)

As \( \omega \) increases between 0 and \( \omega_c \), \( |H(\omega)| \) decreases from 1 to \( 1 / T_{2M} \left( 1 / \cos(\omega_c / 2) \right) \). As \( \omega \) continues between \( f_c \) and \( \pi \), \( |H(\omega)| \) oscillates between \( 1 / T_{2M} \left( 1 / \cos(\omega_c / 2) \right) \) and \(-1 / T_{2M} \left( 1 / \cos(\omega_c / 2) \right) \). An example of \( |H(\omega)| \) for \( \omega_c = 1 \) and \( N = 201 \) is shown on figure 58.

This function could be a desired magnitude response of a low pass filter, although it decreases very quickly. Even though this is difficult to see on the graph, \( \omega_c \) is the frequency at which \( H(\omega) \) reaches its stop band. The oscillations in the stop band are extremely small. At \( \omega_c = 1 \) and \( M = 100 \), \( 1 / T_{2M} \left( 1 / \cos(\omega_c / 2) \right) = 8.71 \times 10^{-46} \).

We can compute a low pass filter \( a(k) \) – the Dolph-Chebychev filter – from \( |H(\omega)| \) by taking its generalized inverse DFT as above. In closed form, the filter would be as follows.

![Figure 58. The Dolph-Chebychev magnitude response](image-url)
\[ a(k) = \frac{1}{(2M + 1)T_{2M}} \left( \sum_{n=-M}^{M} \left( T_{2M} \left( \frac{\cos \left( \frac{\omega_k}{2} \right)}{\cos \left( \frac{\omega_c}{2} \right)} \right) \cos \left( \frac{2\pi n (k - M)}{2M + 1} \right) \right) \right) \] (12.47)

where \( k = 0, 1, ..., N - 1 \), \( N \) is the length of the filter, \( M = (N - 1) / 2 \), and \( \omega_k \) are the frequencies of the \( N \) inverse DFT components between \(-\pi\) and \( \pi \), \( \omega_k = -\pi + 2\pi k / (N - 1) \). The Dolph-Chebychev window \( w(k) \) is simply the Dolph-Chebychev filter scaled to ensure that it peaks at 1.

\[ w(k) = \frac{a(k)}{a\left(\frac{N-1}{2}\right)} \] (12.48)

Suppose that \( N = 201 \), \( M = 100 \), and \( \omega_k = 1 \). The resulting window is shown on figure 59. Suppose that the sampling frequency is 2000 Hz. Take a low pass filter with a cutoff frequency of 318.3 Hz (i.e., \( \omega_k = 318.3 * 2\pi / 2000 = 1 \)). The magnitude response of the resulting filter is shown on figure 60.

Note that, although the Dolph-Chebychev filter has the known stop band attenuation 1 / \( T_{N-1} (1 / \cos (\omega_k / 2)) \), a filter with the Dolph-Chebychev window can have different stop band attenuation, which will depend on the filter’s cutoff frequency and length.

Note also that, in principle, there is no need to link the parameter \( M \) in the Chebychev polynomials \( T_{2M} \) to the number of points \( N \) in the inverse Fourier transform \( (M = (N - 1) / 2) \). With larger \( M \) the resulting filter has a smaller transition band, but worse stop band attenuation. Similarly, there is no need to use \( \omega_k \) in the window that is exactly the cutoff frequency of the filter. A larger \( \omega_c \), for example, will produce a smaller transition band and worse stop band attenuation. This said, we must choose appropriate \( M \) and \( \omega_k \) so that the width of the main lobe of the Dolph-Chebychev window is sufficiently large, so that the sides of the window do not depress too much of the energy of the filter.