EE123 Course Notes

Anmol Parande

Spring 2020 - Professor Miki Lustig

Disclaimer: These notes reflect EE123 when I took the course (Spring 2020). They may not accurately reflect current course content, so use at your own risk. If you find any typos, errors, etc, please raise an issue on the GitHub repository.

Contents

1 The DFT

Whereas the CTFT takes a continuous signal and outputs a continuous frequency spectrum and the DTFT takes a discrete signal and outputs a continuous, periodic frequecy spectrum, the Discrete Fourier Transform takes a discrete finite signal and outputs a discrete frequency spectrum. This is useful for signal processing because we cannot store infinite signals in a computers memory.

Definition 1 *For a length N finite sequence* $\{x[n]\}_0^{n-1}$*, the Discrete Fourier Transform of the signal is a length N finite sequence* $\{X[k]\}_0^{n-1}$ *where*

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}$$

One way to interpret the DFT is in terms of the Fourier series for a disrete periodic signal $\tilde{x}[n] = x[((n))_N]$. Recall that the coefficient of the kth term of the Fourier Series is

$$a_k = \frac{1}{N}\sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}$$

Notice that the a_k of the Fourier Series are the DFT values except scaled by a factor of N. This gives an intuitive inverse DFT.

Definition 2 *For a length N finite sequence* $\{X[k]\}_0^{N-1}$ *representing the DFT of a finite perioidc signal* $\{x[n]\}_0^{N-1}$*, the inverse DFT is given by*

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j\frac{2\pi}{N}kn}$$

Notice that the DFT and the IDFT are very similar in form. It turns out that the IDFT can be expressed as a DFT of $X^*[k]$. Namely

$$IDFT\{X[k]\} = \frac{1}{N}DFT\{X^\star[k]\}^\star$$

Further intuition for the DFT comes from relating it to the DTFT. Suppose we have a finite signal $x[n]$ which is 0 for $n < 0$ and $n > N-1$. The DTFT of this signal is

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} = \sum_{n=0}^{N-1} x[n]e^{-j\omega n}$$

Suppose we sample the DTFT at intervals of $\frac{2\pi}{N}k$, then

$$X[k] = X\left(\frac{2\pi}{N}k\right) = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}$$

Thus we can think of the DFT as a N point sample of the DTFT. One important point to notice though is that while the DTFT is often centered around 0, because we are summing from 0 to N-1 in the DFT, the DFT coefficients are centered around π.

1.1 Convolution and the DFT

1.1.1 Circular Convolution

When the DFT coefficients of two signals are multiplied, the resulting coefficients describe a circular convolution of the original two signals.

$$x[n] \circledast y[n] \leftrightarrow X[k]Y[k]$$

The circular convolution is defined as follows

$$x[n] \circledast y[n] = \sum_{m=0}^{N-1} x[m]y[((n-m))_N]$$

The mechanics of the circular convolution are the same as that of the regular convolution, except the signal is circularly shifted. A circular convolution is

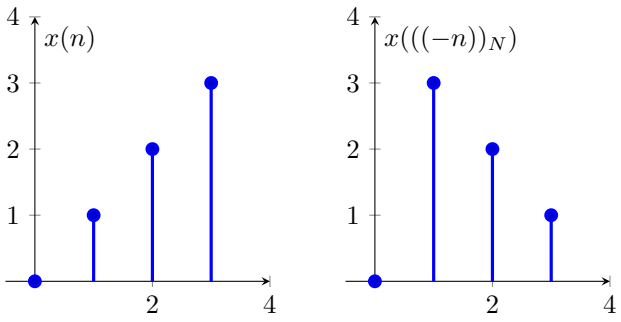

Figure 1: A circular shift

equivalent to a periodic convolution over a single period.

1.1.2 Linear Convolution with the DFT

Because multiplying DFT coefficients performs a convolution, it turns out that we can compute a linear convolution using the circular convolution.

$$\begin{aligned} x[n] \quad & 0 \leq n \leq L-1 \\ h[n] \quad & 0 \leq n \leq P-1 \end{aligned}$$

The linear convolution of these two signals will be length $L+P-1$, so in order to take an IDFT and get $L+P-1$ samples, we need to take at least $N \leq L+P-1$ points.

1. Pad each vector to length $L+P-1$
2. Compute $X[k]H[k]$
3. Take the Inverse DFT

If N is too small, the result is akin to aliasing in the time domain. To see why, consider that the DFT coefficients are essentially the DFS coefficients of the periodic extension of $x[n]$

$$\tilde{x}[n] = \sum_{r=-\infty}^{\infty} x[n - rN]$$

If we compute the DTFT of each periodic extension, then

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega})$$

and the IDTFT of this will be

$$\tilde{y}[n] = \sum_{r=-\infty}^{\infty} y[n - rN]$$

Notice that if N is not large enough, then these copies will be overlapping (a.k.a aliasing). Since the DFT is just sampling the DTFT, the circular convolution will represent the true convolution so long as the copies don't overlap.

1.1.3 Block Convolutions

In a discrete time system, the input signal might have a very long length, making it impractical to be stored in a computer's memory or to compute the DFT of it all at once (especially if we have a real-time system). To compute the output of the filter (with impulse response of length P), we need to compute the DFT in blocks shorter than the signal.

The first method of block convolution is the overlap-add method.

1. Decompose $x[n]$ into nonoverlapping segments of length L

$$x_r[n] = \begin{cases} x[n] & rL \leq n \leq (r+1)L \\ 0 & \text{else} \end{cases}$$

$$x[n] = \sum_r x_r[n]$$

2. Since convolution is linear

$$y[n] = x[n] * h[n] = \sum_r x_r[n] * h[n]$$

3. Zero pad $x_r[n]$ and $h[n]$ to length $N \geq L + P - 1$

4. Compute the DFTs, multiply them, and take the inverse.

5. The neighboring outputs overlap $P - 1$, add the overlapping sections together to get the final output

The other method of block convolution is the overlap-save method.

1. Divide $x[n]$ into sections of length L such that each section overlaps the previous by $P-1$ points

$$x_r[n] = x[n + r(L - P + 1) - P + 1] \qquad 0 \leq n \leq L - 1$$

2. Zero pad $x_r[n]$ and $h[n]$ to length $N \geq L + P - 1$

3. Compute the DFTs, multiply the coefficients, and compute the inverse.

4. The first $P-1$ samples of the output will be incorrect, so we can discard them.

$$y[n] = \sum_{r=0}^{\infty} y_r[n - r(L - P + 1) + P - 1]$$

$$y_r[n] = \begin{cases} x_r[n] * h[n] & P - 1 \leq n \leq L - 1 \\ 0 & \text{else} \end{cases}$$