OFDM and Cyclic Prefix - a hands-on demonstration using Matlab

Mathuranathan Viswanathan
http://www.gaussianwaves.com

May 26, 2016

1 Introduction

2 Introduction

A OFDM transceiver architecture is typically implemented using IFFT and FFT blocks. In a OFDM transmitter, the modulated information symbols are assigned to invidual subcarriers, forming a single OFDM symbol in frequency domain. The OFDM symbol in frequency domain is converted to time domain using IFFT block. The time domain samples are then transmitted across a channel that is modeled using Channel Impulse Response (CIR). On the other-hand, the receiver converts the time domain samples of the received OFDM symbols and converts them to frequency domain for further demodulation of the information in the individual carriers.

In reality, a multipath channel or any channel in nature acts as a *linear* filter on the transmitted OFDM symbols. Mathematically, the transmitted OFDM symbol (denoted as $s[n]$) gets *linearly* convolved with the CIR $h[n]$ and gets corrupted with additive white gaussian noise - designated as $w[n]$. Denoting *linear convolution* as '$*$' and using 'N'-point DFT, received signal in discrete-time can be represented as

$$r[n] = h[n] * s[n] + w[n] \quad (* \rightarrow linear\ convolution) \tag{1}$$

The idea behind using OFDM is to combat frequency selective fading , where different frequency components of a transmitted signal can undergo different levels of fading. The OFDM divides the available spectrum in to small chunks called sub-channels. Within the subchannels, the fading experienced by the individual modulated symbol can be considered flat. This opens-up the possibility of using a simple frequency domain equalizer to neutralize the channel effects in the individual subchannels.

3 Circular convolution and designing a simple frequency domain equalizer

From one of the DFT properties, we know that the *circular convolution* of two sequences in time domain is equivalent to the multiplication of their individual responses in frequency domain.

Let $s[n]$ and $h[n]$ are two sequences of length N with their DFTs denoted as $S[k]$ and $H[k]$ respectively. Denoting *circular convolution* as $\circledast$,

$$h[n] \circledast s[n] \quad \equiv H[k]S[k] \tag{2}$$

If we ignore channel noise in the OFDM transmission, the received signal is written as

$$r[n] = h[n] * s[n] \qquad (* \rightarrow linear\ convolution) \tag{3}$$

Note that the channel performs a linear convolution operation on the transmitted signal. Instead, if the channel performs circular convolution, then the equation would have been

$$r[n] = h[n] \circledast s[n] \qquad (* \rightarrow \textit{circular convolution}) \tag{4}$$

Then, by applying the DFT property given in equation 2,

$$r[n] = h[n] \circledast s[n] \quad \equiv R[k] = H[k]S[k] \tag{5}$$

As a consequence, the channel effects can be neutralized at the receiver using a simple frequency domain equalizer (actually this is a zero-forcing equalizer when viewed in time domain) that just inverts the *estimated* channel response and multiplies it with the frequency response of the received signal to obtain the estimates of the OFDM symbols in the subcarriers as

$$\hat{S}[k] = \frac{R[k]}{H[k]} \tag{6}$$

4 Demonstrating the role of Cyclic Prefix

The simple frequency domain equalizer shown in equation 6 is possible only if the channel performs circular convolution. But in nature, all channels perform linear convolution. The linear convolution can be converted into circular convolution by adding Cyclic Prefix (CP) in the OFDM architecture. Let's understand this by demonstration.

To simply stuffs, we will create two randomized vectors for $s[n]$ and $h[n]$. $s[n]$ is of length 8, the channel impulse response $h[n]$ is of length 3 and we intend to use $N = 8$-point DFT/IDFT wherever applicable.

```
N=8; %period of DFT
s = randn(1,8);
h = randn(1,3);
```

```
>> s =   -0.0155    2.5770    1.9238   -0.0629   -0.8105    0.6727   -1.5924   -0.8007
>> h =   -0.4878   -1.5351    0.2355
```

Now, convolve the vectors **s** and **h** linearly and circularly. The outputs are plotted in Figure 1. We note that the linear convolution and the circular convolution does not yield identical results.

```
lin_s_h = conv(h,s) %linear convolution of h and s
cir_s_h = cconv(h,s,N) % circular convolution of h and s with period N
```

```
lin_s_h = 0.0076 -1.2332 -4.8981 -2.3158  0.9449  0.9013 -0.4468  2.9934  0.8542 -0.1885
cir_s_h = 0.8618 -1.4217 -4.8981 -2.3158  0.9449  0.9013 -0.4468  2.9934
```

Let's append a cyclic prefix to the signal **s** by copying last N_{cp} symbols from **s** and pasting it to its front. Since the channel delay is 3 symbols (CIR of length 3), we need to add *atleast* 2 CP symbols.

```
Ncp = 2; %number of symbols to copy and paste for CP
s_cp = [s(end-Ncp+1:end) s]; %copy last Ncp symbols from s and prefix it.
```

```
s_cp = -1.5924 -0.8007 -0.0155  2.5770  1.9238 -0.0629 -0.8105  0.6727 -1.5924 -0.8007
```

Lets assume that we send the cyclic-prefixed OFDM symbol $\mathbf{s_{cp}}$ through the channel which perform linear filtering

```
lin_scp_h = conv(h,s_cp) %linear convolution of CP-OFDM symbol s_cp and the CIR h
```

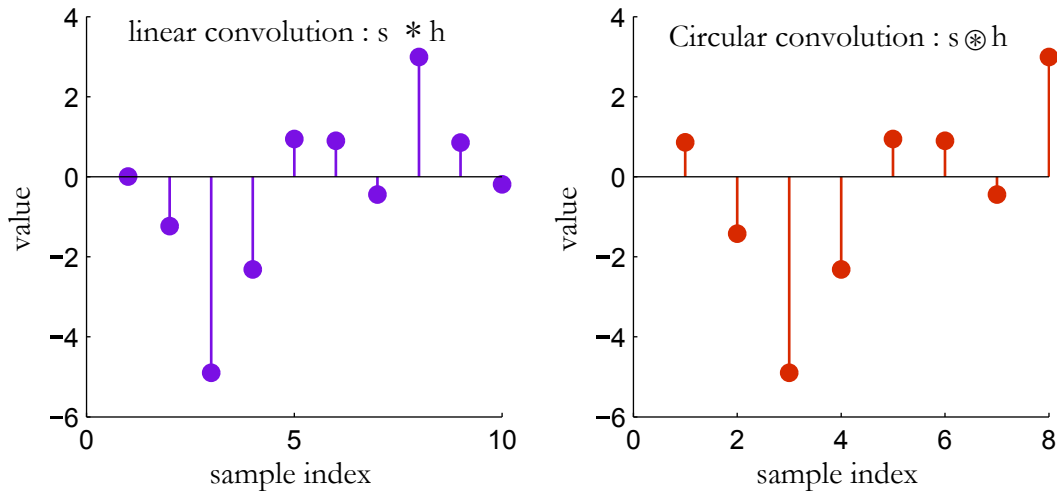

Figure 1: Difference between linear convolution and circular convolution

```
lin_scp_h = 0.7767  2.8350  0.8618 -1.4217 -4.8981 -2.3158  0.9449  0.9013 -0.4468  2.9934
    0.8542 -0.1885
```

Compare the outputs due to `cir_s_h` $= \mathbf{s} \circledast \mathbf{h}$ and `lin_scp_h` $= \mathbf{s}_{cp} * \mathbf{h}$. We can immediately recognize that that the middle section of `lin_scp_h` is exactly same as the `cir_s_h` vector. This is shown in the figure 2.

```
cir_s_h   = 0.8618  -1.4217 -4.8981 -2.3158  0.9449  0.9013 -0.4468  2.9934
lin_scp_h = 0.7767   2.8350  0.8618 -1.4217 -4.8981 -2.3158  0.9449  0.9013 -0.4468 2.9934
    0.8542 -0.1885
```

So far we have faked the channel to perform circular convolution by adding a cyclic extension to the OFDM symbol. We are not done yet. At the receiver, we are only interested in the the middle section of the `lin_scp_h` which is the channel output. The first block in the OFDM receiver removes the additional symbols from the front and back of the channel output. This becomes the received symbol **r** after the removal of cyclic prefix in the receiver.

```
r = lin_scp_h(Ncp+1:N+Ncp)%cut from index Ncp+1 to N+Ncp
```

5 Verifying DFT property

The DFT property given in equation 5 can be re-written as

$$\begin{aligned} r[n] &= IDFT\{R[k]\} \\ &= IDFT\{H[k]S[k]\} \\ &= IDFT\{DFT(h[n])\,DFT(s[n])\} \\ &= h[n] \circledast s[n] \end{aligned} \tag{7}$$

To verify this in Matlab, take N-point DFTs of the received signal and CIR. Then, we can see that the IDFT of product of DFTs of **s** and **h** will be equal to the N-point circular convolution of **s** and **h**

```
R = fft(r,N); %frequency response of received signal
H = fft(h,N); %frequency response of CIR
S = fft(s,N); %frequency response of OFDM signal (non CP)
```

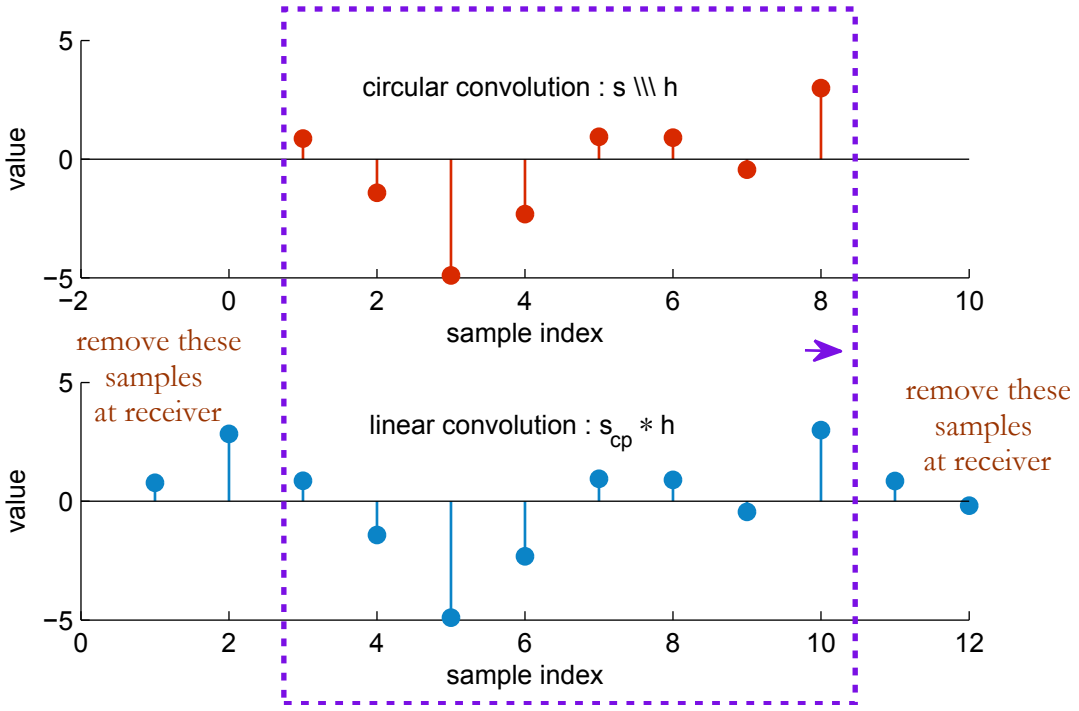

Figure 2: Equivalence of circular convolution and linear convolution with cyclic prefixed signal

```
r1 = ifft(S.*H); %IFFT of product of individual DFTs

display(['IFFT(DFT(H)*DFT(S)) : ',num2str(r1)])
display([cconv(s,h): ', numstr(r)])
```

```
IFFT(DFT(H)*DFT(S)) : 0.86175  -1.4217  -4.8981  -2.3158  0.94491  0.90128 -0.44682  2.9934
cconv(s,h):           0.86175  -1.4217  -4.8981  -2.3158  0.94491  0.90128 -0.44682  2.9934
```