Digital Communication using GNU Radio

Prof. Kumar Appaiah

Department of Electrical Engineering

Indian Institute of Technology Bombay

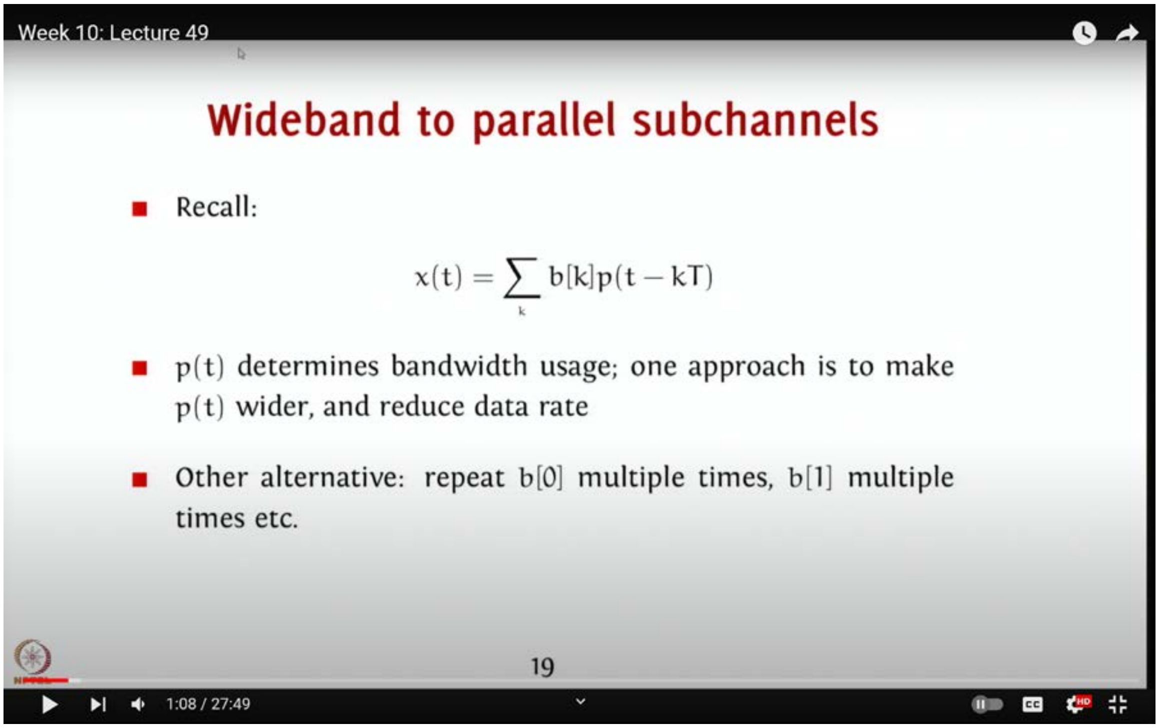
Week-10

Lecture-49

Orthogonal Frequency Division Multiplexing (OFDM)

Hello, and welcome to this lecture on Digital Communication Using GNU Radio. I'm Kumar Appiah from the Department of Electrical Engineering at IIT Bombay. Today, we will be continuing our discussion on Orthogonal Frequency Division Multiplexing (OFDM), picking up where we left off in the previous lecture.

(Refer Slide Time: 01:08)



As you may recall, in our previous discussion, we explored the concept of converting a wideband channel into several parallel narrowband channels. This transformation allows these smaller narrowband channels, referred to as subcarriers, to behave as flat channels, simplifying the equalization process significantly. We will expand on this idea today.

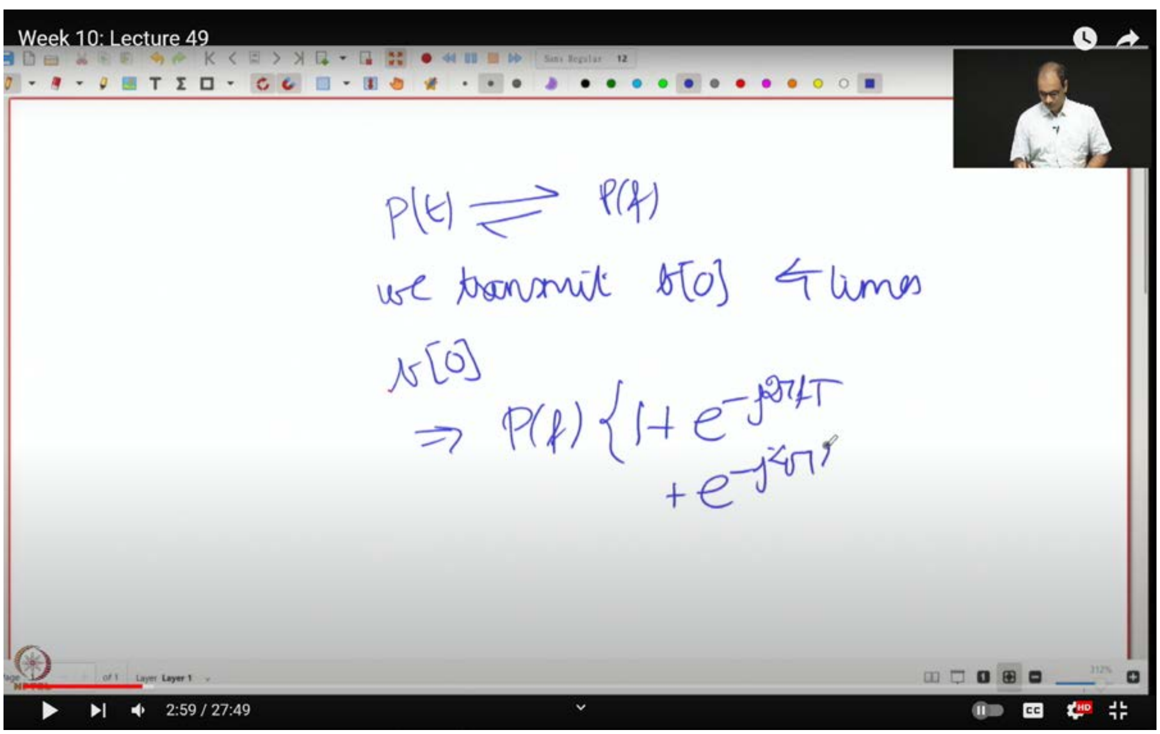
If you remember from our previous lectures, the transmit signal model we're considering is:

$$x(t) = \sum b_k \cdot p(t - kT)$$

where b_k represents the symbols being transmitted, and p(t - kT) denotes the effective pulse as seen by the receiver. This effective pulse is essentially the convolution of the transmit pulse with the channel's impulse response, if present.

The pulse shape p(t) plays a crucial role in determining the bandwidth usage. Since p(t) results from the convolution of $g_{\text{tx}}(t)$ (the transmit pulse shape) and the channel, the bandwidth usage is naturally limited by the narrower of the two bandwidths. Typically, the transmit pulse $g_{\text{tx}}(t)$ is shaped in such a way that it has a constrained bandwidth footprint.

(Refer Slide Time: 02:59)



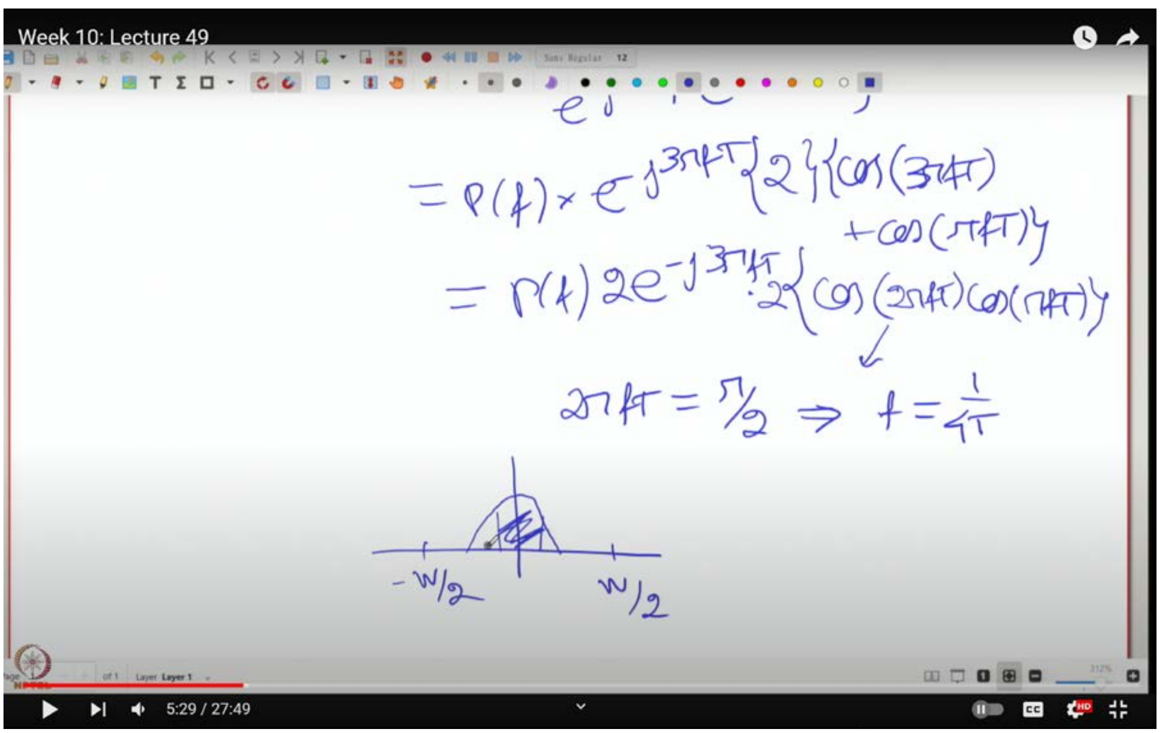
To reduce the channel's frequency domain spread, one approach is to widen the pulse p(t) and subsequently reduce the data rate. This is one way of addressing the problem. Another alternative, as we discussed in the previous lecture, is to repeat symbols like b_0 and b_1 multiple times. If you recall, we explored the scenario where p(t) has a Fourier transform denoted as p(f). Suppose we transmit the symbol b_0 four times. This results in the effective pulse being expressed as:

$$p(t) + p(t - T) + p(t - 2T) + p(t - 3T)$$

We derived this expression during the last class, and today we will build on that foundation to deepen our understanding of OFDM.

Let’s continue our exploration of how these ideas fit together and lead to the powerful technique of OFDM.

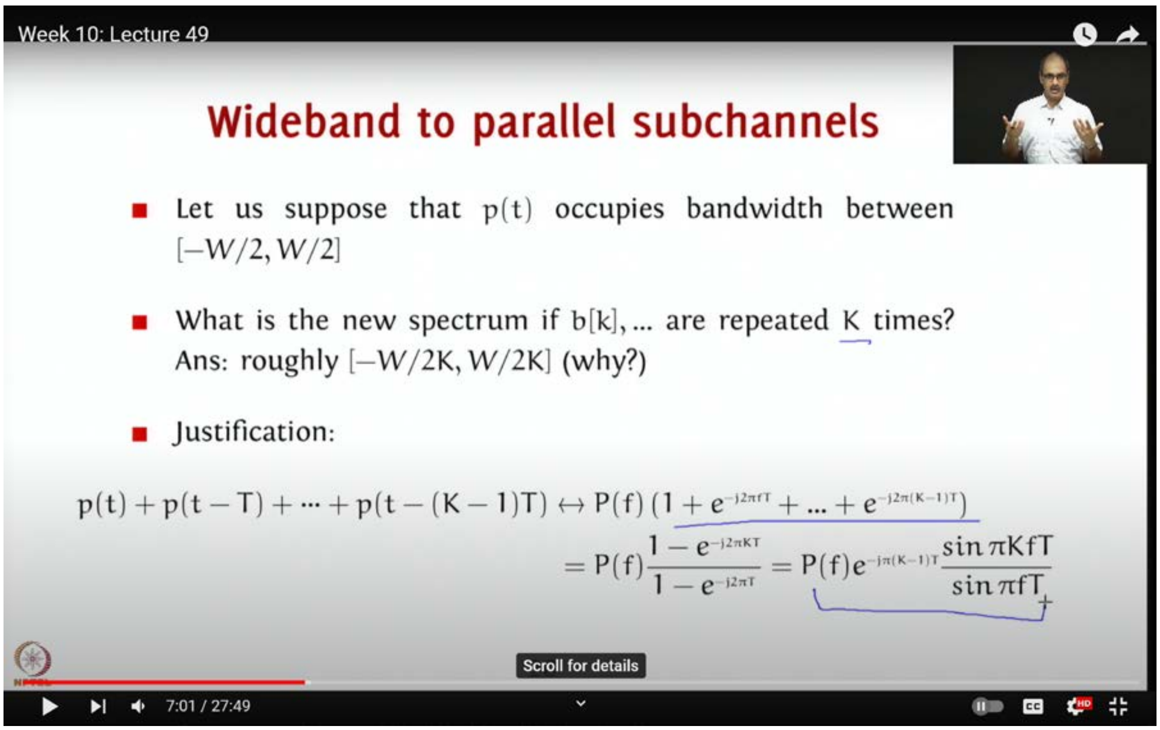
(Refer Slide Time: 05:29)



If you combine these terms carefully, you will get the expression $p(f) \times 2 \times cos(3\pi ft)$, where I have taken $e^{-j3\pi ft}$ out as a common factor. This results in $2 \times cos(3\pi ft) +$

$cos(\pi f t)$. To verify this, if you take $e^{-j3\pi f t}$ as a common factor, you'll get terms like $e^{j3\pi f t}$ and $e^{-j3\pi f t}$, and when you add them, you'll end up with $2cos(3\pi f t)$. Similarly, for the other term, you'll have $e^{j\pi f t} and e^{-j\pi f t}$, which will combine to give $cos(\pi f t)$.

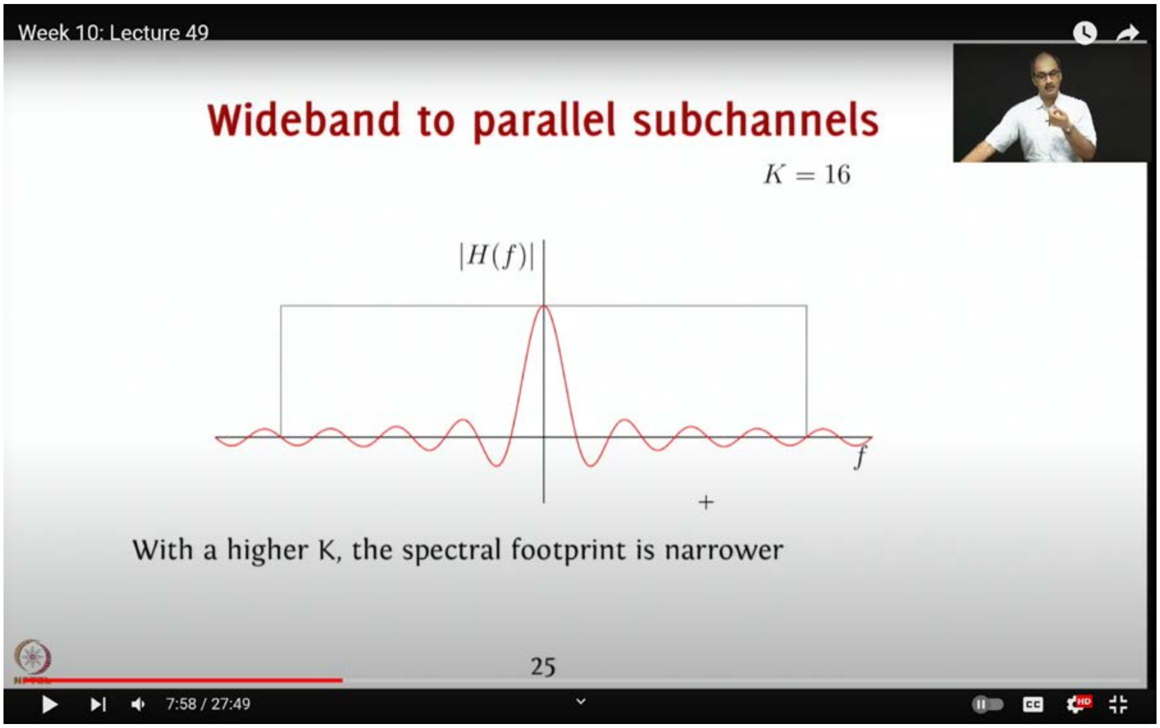
(Refer Slide Time: 07:01)



Now, when you add these cosine terms, applying the trigonometric identity for the sum of cosines, you'll have $2cos\left(\frac{4\pi f t}{2}\right)cos\left(\frac{2\pi f t}{2}\right)$, which simplifies to $2cos(2\pi f t)cos(\pi f t)$. Now, if you examine the first cosine term $cos(2\pi f t)$, it becomes zero at values of f such that $2\pi f t = \pm\frac{\pi}{2}$. This leads to the condition $f = \frac{1}{4\mathrm{T}}$.

To better visualize this, imagine plotting the spectrum. Let's assume that the range is from $-\frac{1}{2t}$ to $\frac{1}{2t}$, which can also be expressed as $-\frac{w}{2}$ to $\frac{w}{2}$, since $t = \frac{1}{w}$. With this setup, the spectrum will drop to zero at specific points, effectively reducing the bandwidth usage to this smaller region. What this means is that you are now using approximately one-fourth of the original bandwidth. This reduction in bandwidth aligns with our intuitive understanding from digital signal processing (DSP): slowing down the pulse in the time domain, whether by repeating symbols or other methods, naturally compresses the bandwidth in the frequency domain.

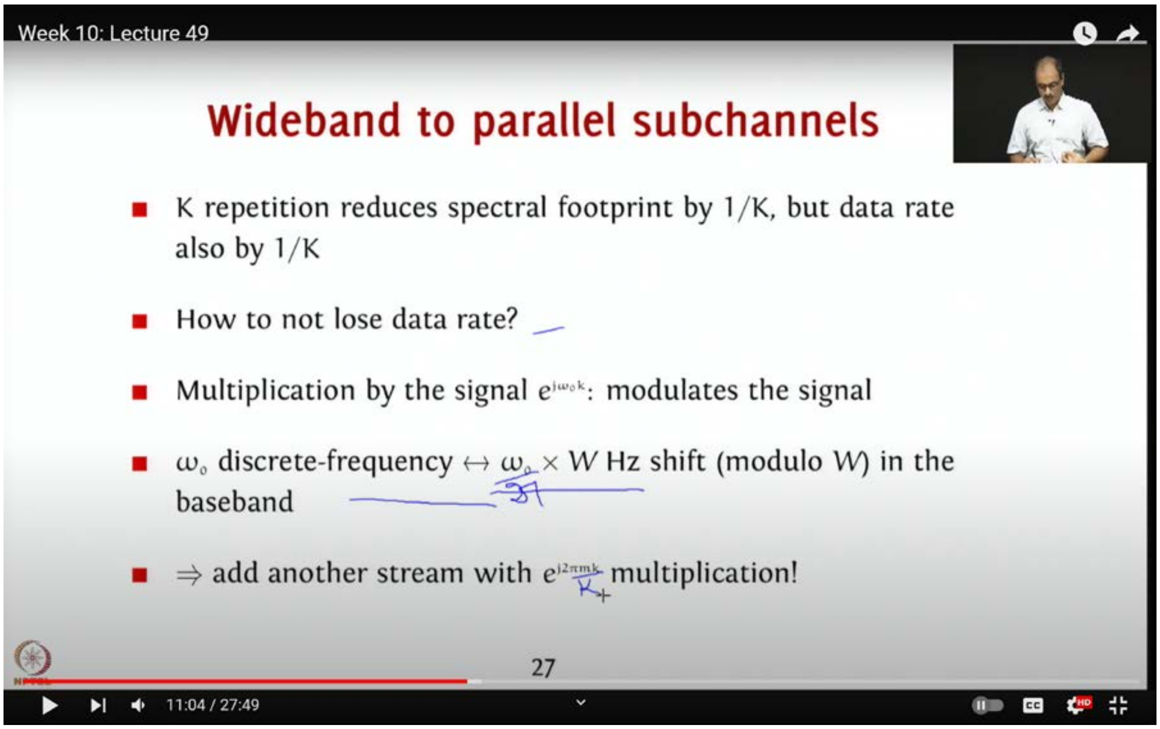
(Refer Slide Time: 07:58)



The key idea here is that we are repeating symbols like b_0 multiple times, as well as b_1, perhaps repeating them k times each. While this repetition helps to flatten the channel, narrowing the frequency spectrum usage, it comes at the cost of a reduced data rate. Initially, we were transmitting one symbol every t seconds, but now, after repeating the symbols, we are transmitting one symbol every kt seconds. For example, if k = 4, we would be sending one symbol every 4t seconds. This approach severely impacts the data rate, which is clearly not ideal.

So, what's a better approach? Let's suppose that the pulse p(t) occupies the bandwidth between $-\frac{w}{2}$ and $\frac{w}{2}$. This is what we have been discussing so far. However, if you now repeat the symbol b_k multiple times, let's say K times, then the new spectrum will be reduced to a range of approximately $-\frac{w}{2K}$ to $\frac{w}{2K}$. This is justified because if you repeat $p(t) + p(t-t) + \cdots + p(t-(k-1)t)$, the resulting effective spectrum will be narrower, as each delay introduces a phase shift of $e^{-j2\pi ft}$. The accumulation of these phase shifts compresses the spectrum.

(Refer Slide Time: 11:04)



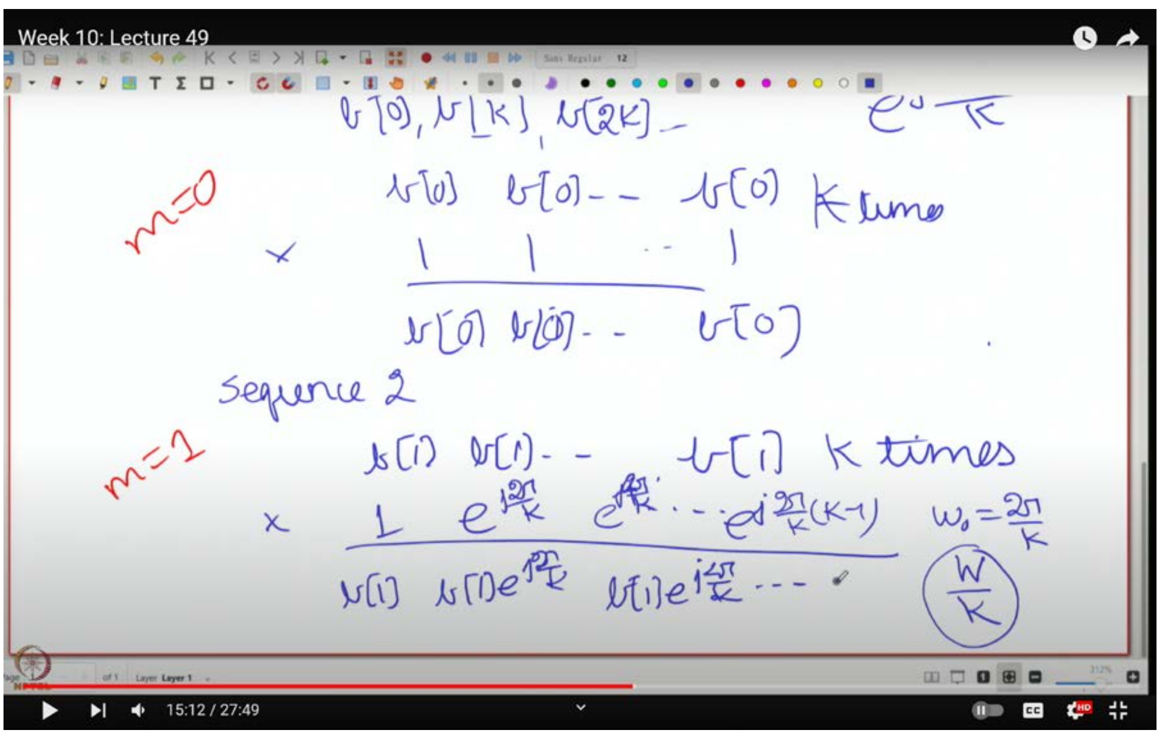
If you account for all the terms and perform a geometric series summation, you'll arrive at something that resembles a sinc function, which makes intuitive sense. As k becomes larger and larger, the spectrum becomes progressively narrower. The downside, however, is that the data rate also reduces to $\frac{1}{k}$. This is what we've observed.

We tried this with k = 4. When k = 2, the spectral footprint changes, and you get a spectrum corresponding to repeating b_0 and b_1 two times each, and so forth. For k = 4, you see a different spectral footprint. Recall that when the bandwidth was originally between $-\frac{w}{2}$ and $\frac{w}{2}$, after repeating the symbols four times, the spectrum was reduced to $\frac{w}{4}$, which corresponds to $\frac{1}{4t}$. As k increases, the pulses become narrower and start resembling sinc functions more closely. This behavior aligns well with expectations, given the Fourier transform properties.

Now, the issue we face is that while the spectral footprint becomes narrower, the data rate is reduced. How do we resolve this problem? The key idea here is that repeating the symbols k times reduces the spectral footprint by a factor of $\frac{1}{k}$, but it also reduces the data rate by the same factor, $\frac{1}{k}$. To counteract this, instead of simply reducing the data rate, we introduce another parallel stream and multiply it by $e^{j\omega_0 k}$.

What we're doing is reducing the spectral footprint of the signal, but then we take another pulse sequence with the same reduced spectral footprint and place it in a neighboring frequency band where it doesn't cause interference. The multiplication by $e^{j\omega_0 k}$ modulates the signal, this is something familiar from digital signal processing (DSP). In the Discrete-Time Fourier Transform (DTFT) domain, multiplying by $e^{j\omega_0 k}$ shifts the spectrum by ω_0 to the right.

(Refer Slide Time: 15:12)



So, even though you've reduced the bandwidth of the pulse, which has cut the data rate by $\frac{1}{k}$, you can take another narrow pulse with the same bandwidth and modulate it by $e^{j\omega_0 k}$.

This modulation shifts the spectrum of the new pulse to a neighboring frequency band, keeping it separate from the previous pulse. You can continue this process with additional streams, placing them side by side in the frequency domain. Essentially, you are slicing your available bandwidth into smaller bins and filling those bins with these narrow-band sequences, all running in parallel.

The key concept here is the use of ω_0, which represents a discrete frequency. When you multiply by $e^{j\omega_0 k}$, the frequency shift in hertz depends on ω_0. From your DSP knowledge, you know that you are using a frequency range from $-\frac{w}{2}$ to $\frac{w}{2}$, and this type of signaling, whether it's sinc or a root-raised cosine, satisfies the Nyquist ISI-free criterion. Recall that $\frac{w}{2}$ is analogous to $\frac{f_s}{2}$, where f_s is the sampling frequency. So, when you modulate by $e^{j\omega_0 k}$, the frequency shift in hertz will be $\omega_0 \times w$. So, this method allows you to maintain the same overall bandwidth usage while placing multiple streams in parallel, which solves the problem of reduced data rate without expanding the total spectral footprint.

In other words, when you select ω_0, it should include a 2π multiple. For instance, if you choose $\omega_0 = \pi$, that corresponds to a shift of $\frac{w}{2}$ in the frequency domain. Similarly, if you choose $\omega_0 = \frac{\pi}{4}$, the shift would be around $\frac{w}{8}$. Now, these frequency shifts essentially exist modulo w, as your operating range is between $-\frac{w}{2}$ and $\frac{w}{2}$. The exact division of the frequency spectrum depends on multiplying by $e^{j\frac{2\pi mk}{K}}$.

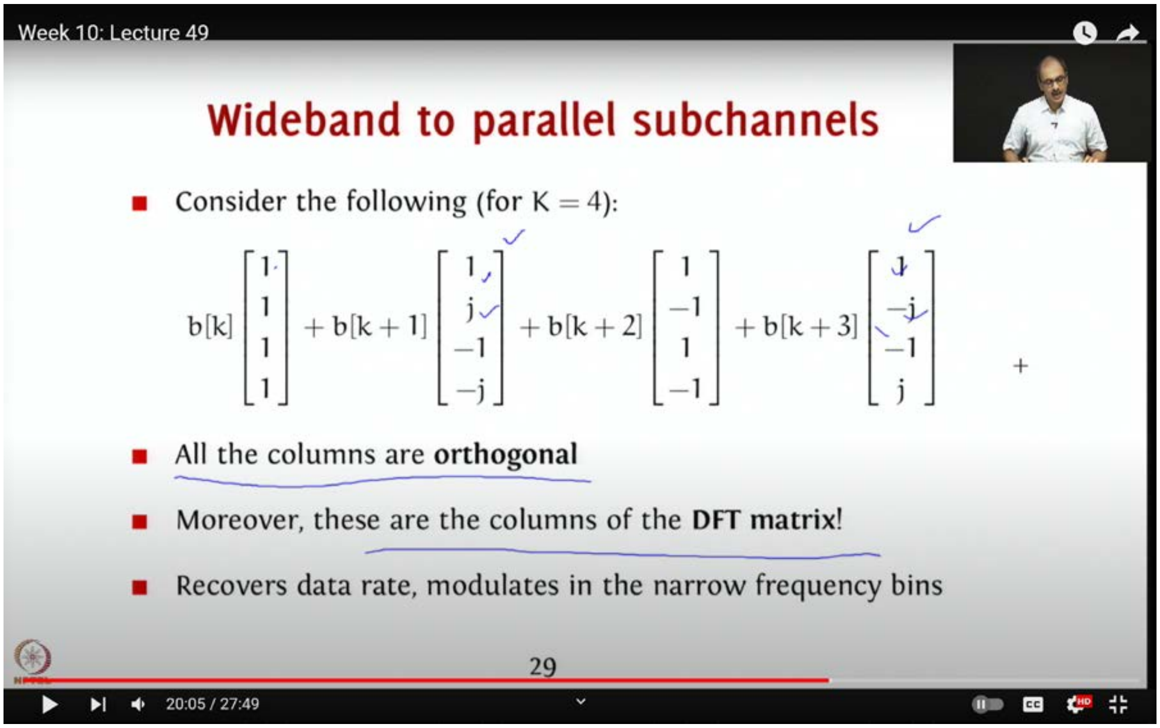
Here's how it works: you select m as a number between $0, 1, 2, \ldots, k-1$. So m will take values from 0 up to k-1. Let me break this down step by step.

What we're essentially doing is taking the first sequence of data, let's say you have b_0, b_k, b_{2k}, and so on. You group these symbols, and then you repeat them k times. So, for the first sequence, you'll have something like b_0, b_0, and up to b_{0k}. This repeated sequence will then be multiplied by $e^{j\frac{2\pi mk}{K}}$, where m takes values from 0 to k-1.

For the first sequence, let's say we choose m = 0, so you multiply by e^{j0}. In this case, you'll have b_0, b_0, b_0, repeated up to k times. This corresponds to m = 0.

Now let's take the second sequence, which corresponds to m = 1. In this case, we have b_1, b_1, repeated k times. For this sequence, you multiply by $e^{j\frac{2\pi}{k}m}$, so you'll have $e^{j0}, e^{j\frac{2\pi}{k}}, e^{j\frac{4\pi}{k}}, \ldots, e^{j\frac{2\pi(k-1)}{k}}$, and you send this modified sequence.

(Refer Slide Time: 20:05)



In simpler terms, for sequence 2, you'll multiply by $e^{j\frac{2\pi mk}{K}}$, which shifts the frequency spectrum by $\omega_0 = \frac{2\pi}{k}$. In hertz, this corresponds to a frequency shift of $\frac{W}{k}$. Thus, when you divide the sequence and repeat it k times, the spectral footprint becomes $\frac{1}{k}$ of the original. However, by multiplying by $e^{j\omega_0 n}$ or $e^{j\omega_0 k}$, where $\omega_0 = \frac{2\pi}{k}$, you shift the spectrum by $\frac{W}{k}$, moving it into a neighboring frequency band.

If you then take sequence 3, you would repeat the process again, shifting the spectrum further into the next adjacent frequency band. Another way to view this is through vectors. For instance, you could consider a vector of all ones, representing no spectral shift. Then, for sequence 2, you multiply by $e^{j\frac{2\pi}{k}}$, giving you a vector of 1, $e^{j\frac{2\pi}{k}}, e^{j\frac{4\pi}{k}}$, ..., effectively shifting the sequence's spectrum.

These vectors are actually orthogonal. This means that if you take the sequence b_0, b_0, b_0, b_0 and add it to the sequence $b_1, b_1 \cdot e^{j\frac{2\pi}{k}}, b_1 \cdot e^{j\frac{4\pi}{k}}$, ..., the resulting vector will allow you to recover b_0 and b_1 by simply taking the inner product with another set of orthogonal vectors. This insight should give you a clearer understanding of how we approach this problem using orthogonal vector properties.

Let's now take a closer look at the case where k = 4. For this scenario, we take the block $b_k, b_{k+1}, b_{k+2}, b_{k+3}$, and repeat each element four times, placing it into a column vector. For example, you would have something like b_k, b_k, b_k, b_k. Before adding the repeated b_{k+1} sequence to this, we multiply it by the terms $1, e^{j\frac{2\pi}{4}}, e^{j\frac{4\pi}{4}}, e^{j\frac{6\pi}{4}}$. This multiplication is governed by the formula $e^{j\frac{2\pi mk}{4}}$, where m ranges from 0 to 3.

So, for m = 0, you multiply by e^{j0}, while for m = 1, you multiply by $e^{j\frac{2\pi}{4}}, e^{j\frac{4\pi}{4}}, e^{j\frac{6\pi}{4}}$, and so on. Evaluating these terms yields,

$$e^{j\frac{2\pi}{4}} = e^{j\frac{\pi}{2}} = j, e^{j\frac{4\pi}{4}} = e^{j\pi} = -1, e^{j\frac{6\pi}{4}} = e^{j\left(-\frac{\pi}{2}\right)} = -j.$$

So, for this sequence, you obtain the vector 1, j, -1, -j.

Next, let's take the block for b_{k+2}, which repeats similarly. The values for the exponential terms follow the pattern: $e^{j\frac{4\pi}{4}} = -1, e^{j\frac{8\pi}{4}} = e^{j2\pi} = 1, e^{j\frac{12\pi}{4}} = e^{j3\pi} = -1$, and so forth. When you repeat this for the sequence b_{k+3}, you get terms like $e^{j\frac{6\pi}{4}} = -j, e^{j\frac{12\pi}{4}} = -1$, and so on.

For convenience, I have rewritten the vector forms:

- [1, 1, 1, 1]
- [1, j, -1, -j]
- [1, -1, 1, -1]

- [1, -j, -1, j]

The key takeaway is that all these column vectors are orthogonal to each other. Orthogonal means that if you take the inner product of any two different columns, the result is zero. For example, if you compute the inner product of the first two vectors, you get:

$$1 \times 1 + 1 \times j + 1 \times (-1) + 1 \times (-j) = 0$$

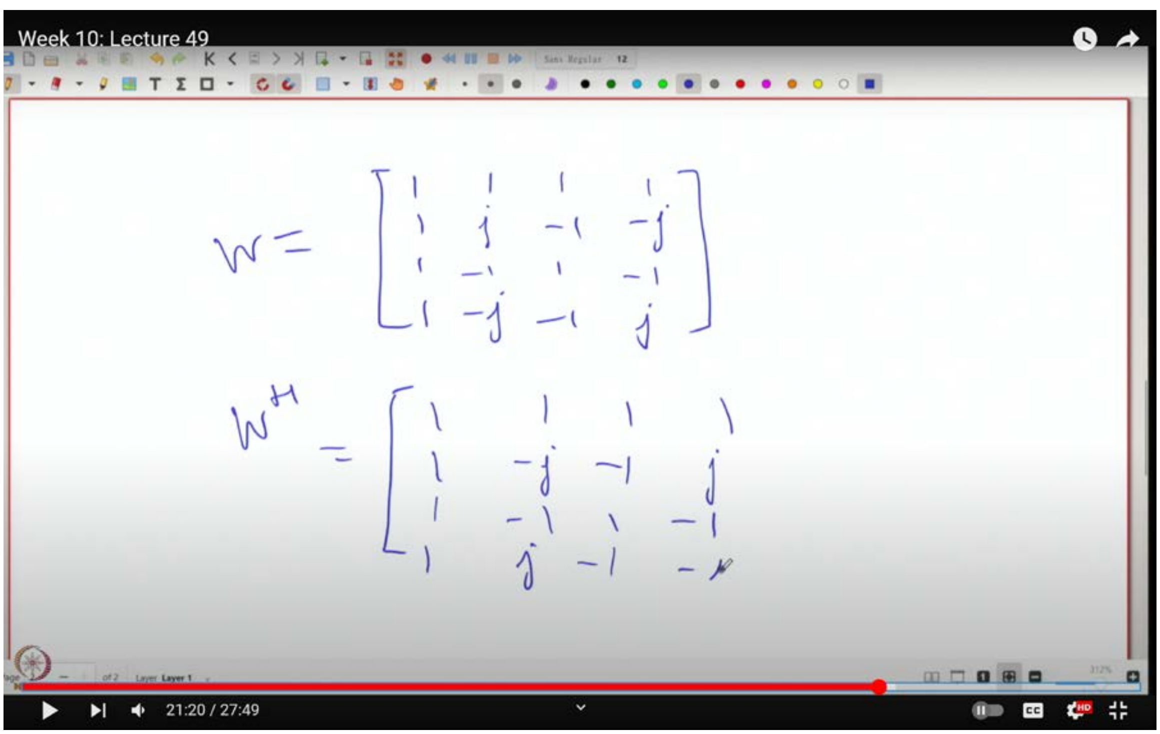
Thus, the vector [1, 1, 1, 1] is orthogonal to [1, j, -1, -j].

You can also verify the orthogonality between other pairs of columns. For instance, take the third and fourth vectors:

$$1 \times 1 + (-1) \times (-j) + 1 \times (-1) + (-1) \times j = 0$$

Therefore, these columns are also orthogonal.

(Refer Slide Time: 21:20)



These columns are, in fact, the columns of the Discrete Fourier Transform (DFT) matrix. By leveraging the DFT matrix, you can parallelize the transmission into multiple narrowband channels, simplifying the process. Interestingly, you don't even need to manually perform the complex multiplication; the DFT operation inherently takes care of this task.

To achieve the desired result, you simply need to utilize the DFT (Discrete Fourier Transform). Keep in mind, though, that when working with inner products, you must remember to take the complex conjugate when necessary. For instance, if you're checking the orthogonality of two complex vectors, like the first and third ones, you'll need to take the conjugate of the second term. For example, $1 \times 1 + j \times (-j)$ with the conjugate of -j becomes j^2, which ensures that the vectors remain orthogonal.

Now, the core idea here is that instead of performing narrowband channel repetition manually, the DFT provides an elegant and straightforward solution to achieve this orthogonality. Let's take a closer look at how this works in practice.

Consider the following matrix, which we'll call W:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{pmatrix}$$

To verify the orthogonality, we need to compute the Hermitian transpose of W, denoted as W^H. The Hermitian transpose involves taking the transpose of the matrix and then taking the conjugate of each entry. For example, the first row of W, which is (1, 1, 1, 1), becomes the first column of W^H, and the second row, (1, j, -1, -j), becomes (1, -j, -1, j) after taking the conjugates. Similarly, for the other rows, you transpose and conjugate accordingly.

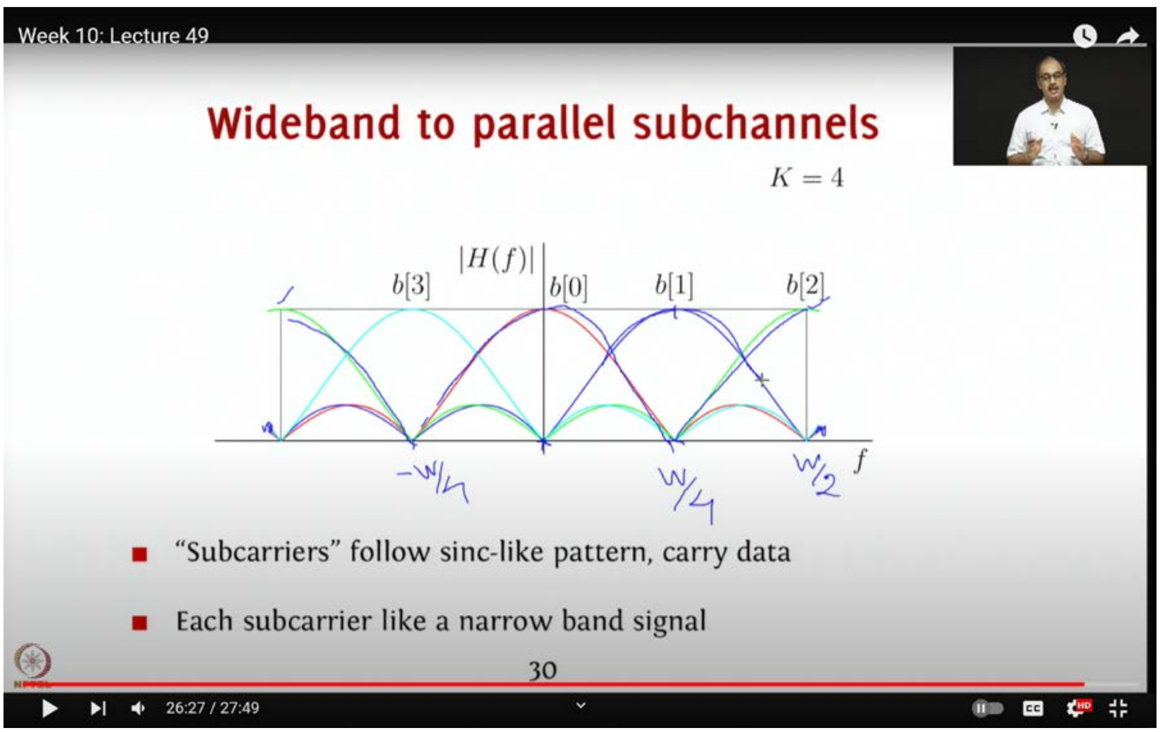
So, W^H would look like this:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{pmatrix}$$

Now, let's perform a basic exercise: calculate $W^H \times W$. If you do the matrix multiplication carefully, you'll notice that the result will yield a diagonal matrix. For example, the first row multiplied by the first column will give you 4, while the first row multiplied by the second column will yield 0. This pattern continues, and you'll end up with zeroes everywhere except on the diagonal, where the values will be non-zero. This is a key property of the DFT matrix, it guarantees that these columns are orthogonal.

However, it's important to note that this DFT matrix is not scaled, which is why W^H W does not result in an identity matrix. Instead, the diagonal entries correspond to the energy in each column. Nonetheless, this is exactly the property of the DFT that we are leveraging to ensure that the transmissions remain orthogonal.

(Refer Slide Time: 26:27)



So what happens in the frequency domain when you apply the DFT? Imagine that each of these repeated narrowband signals gets transformed into the frequency domain. The DFT operation effectively spreads the energy of the time-domain sequence across discrete frequencies. In doing so, it ensures that the signals are sent orthogonally, thereby allowing multiple parallel transmissions without interference between them. This frequency-domain orthogonality is the very essence of what makes DFT-based transmission efficient and robust.

We need to discuss the concept of sub-carriers, which are essentially sinusoidal patterns that carry data. For example, consider the case where k = 4. If we take b_0 and repeat it 4 times, the resulting spectrum exhibits a sinc-like pattern. Specifically, the spectrum is given by

$$\frac{sin(\pi k f_t)}{sin(\pi f_t)}$$

where f_t represents the frequency. The pulse p(t) essentially acts as a window function, shaping the sinc function, and this pulse will cut out the relevant portion of the spectrum.

Now, if we repeat b_1 4 times, it will occupy the same spectral shape as b_0, but it will be shifted. When you multiply b_1 by the vector [1, j, -1, -j], it's equivalent to applying a frequency shift of $\frac{\pi}{2}$ k, which corresponds to a frequency shift of $\frac{w}{4}$. Therefore, if the original center frequency is $\frac{w}{2}$, the new center frequency for b_1 will be $\frac{w}{4}$.

How does orthogonality manifest in this context? We previously established that sequences like [1, 1, 1, 1] and [1, j, -1, -j] are orthogonal. This orthogonality is evident in the frequency domain: at 0 Hz, the b_1 pulses contribute nothing to the spectrum of b_0, and at $\frac{w}{4}$, the b_0 pulses do not influence b_1. Even though we haven't split the channel into rectangular pulses, the orthogonality principle is still respected. This means that b_0 and b_1 are distinguishable from each other without interference from other sequences.

Now, let's consider b_2. In this case, the shift is by π, which, when dealing with frequencies, corresponds to a shift of $\frac{w}{2}$. However, since frequency shifts are cyclic, this $\frac{w}{2}$ shift results in b_2 appearing in two distinct places in the spectrum. In the discrete frequency domain, the Discrete-Time Fourier Transform (DTFT) is periodic with a period of 2π, meaning that shifts by π cause the sequence to repeat itself every 2π radians. Consequently, the frequency components wrap around, causing b_2 to appear at both $-\frac{w}{2}$ and $\frac{w}{2}$.

Finally, consider b_3. The shift for b_3 is by $\frac{3\pi}{2}$, which can also be viewed as $-\frac{\pi}{2}$ or -$\frac{w}{4}$. This shift places b_3 in the frequency domain at $-\frac{w}{4}$.

In summary, you have effectively parallelized the communication channels into four distinct narrow-band channels. These narrow-band channels, characterized by their sinc-like spectral shapes, occupy a much narrower bandwidth compared to the overall range from $-\frac{w}{2}$ to $\frac{w}{2}$. However, by using four parallel streams, each with its own narrow band, the total bandwidth utilized sums up to $\frac{w}{2}$. This is because all four streams are transmitted simultaneously, each using the same time slots in parallel.

It's important to note that the matrix W we discussed is technically the inverse Discrete-Time Fourier Transform (DTFT) matrix, which is used to transform the data into the time domain. We will delve deeper into this in the next lecture. The crucial point here is that the DTFT-based approach facilitates convenient channel parallelization while maintaining orthogonality among the transmissions.

In our upcoming lecture, we will further explore Orthogonal Frequency Division Multiplexing (OFDM), including how it can be effectively used and how the Discrete Fourier Transform (DFT) serves as a useful tool for equalization. Thank you for your attention, and I look forward to continuing our discussion.