Digital Communication using GNU Radio

Prof. Kumar Appaiah

Department of Electrical Engineering

Indian Institute of Technology Bombay

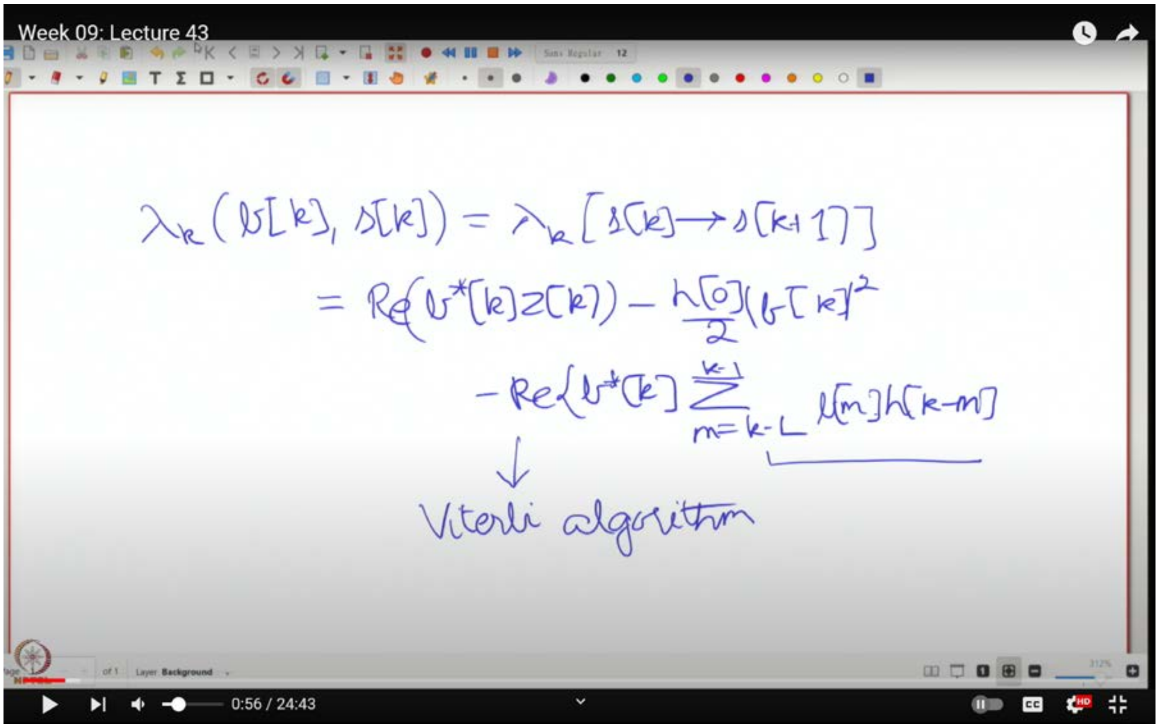
Week-09

Lecture-43

Maximum Likelihood Sequence Estimation: Viterbi Algorithm

Hello, and welcome to this lecture on digital communication using GNU Radio. My name is Kumar Appiah, and I am from the Department of Electrical Engineering at IIT Bombay. In this lecture, we will conclude our discussion of the Viterbi algorithm by working through an example.

(Refer Slide Time: 00:56)

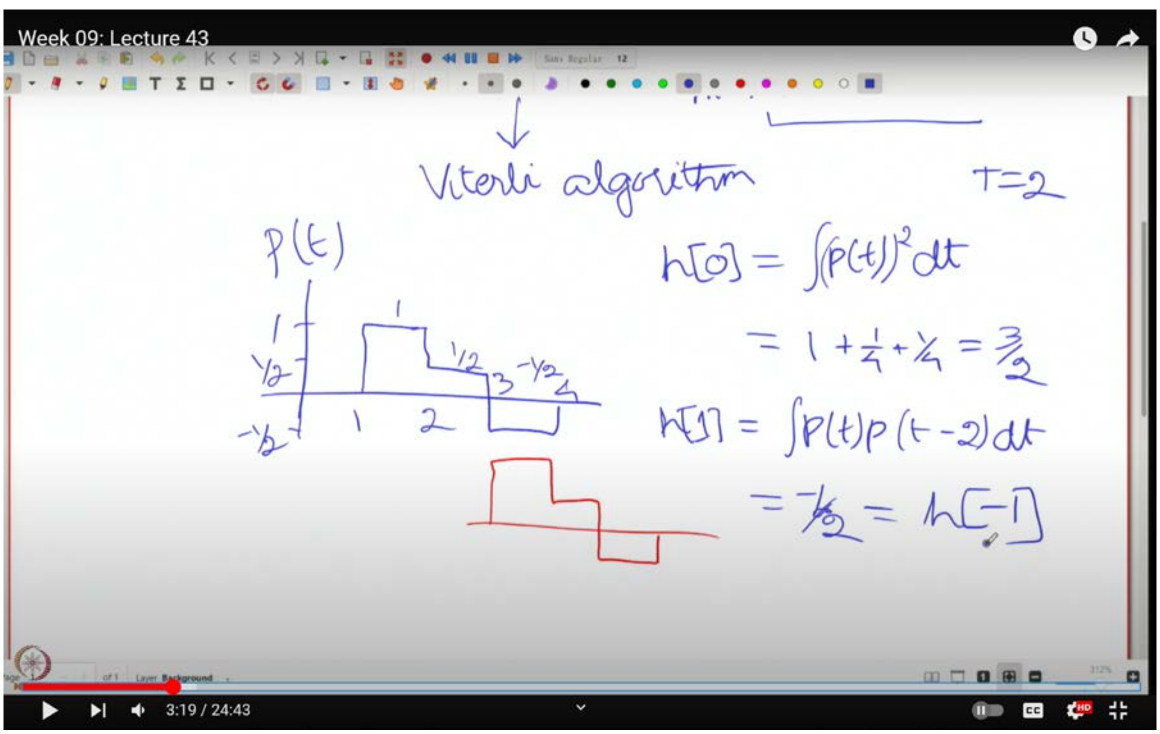


If you recall from the previous lecture, we discussed the concept of an incremental metric, and we claimed that this incremental metric can be used to make decisions in real-time when performing Maximum Likelihood Sequence Estimation (MLSE). The form of the incremental metric was derived from our earlier notation, where Z_k was essentially the result of taking the inner product of Y with S_b at the k-th time instant. Furthermore, the term H represents the autocorrelation of p(t), evaluated at the k-th instant. Now, let us revisit the example we were working on.

In our example, we derived an effective p(t) that had a specific form. Let me remind you of its structure: it had values of 1, 2, 3, and 4, with corresponding values of $-\frac{1}{2}$, $\frac{1}{2}$, and 1. For this function p(t), we will now apply the Viterbi algorithm to a particular example. But before we proceed, we need to compute the corresponding H values: H_0, H_1, H_2, and so on.

Let us first compute H_0. In this case, since everything is real, and we will assume Binary Phase Shift Keying (BPSK) modulation, we can compute H_0 using the integral of p(t) multiplied by $p^*(t)$, i.e., $p(t)^2$. This is straightforward. If you look at the values: 1, $\frac{1}{2}$, and $-\frac{1}{2}$, you get $1+\frac{1}{4}+\frac{1}{4}=\frac{3}{2}$. Hence, $H_0=\frac{3}{2}$.

(Refer Slide Time: 03:19)



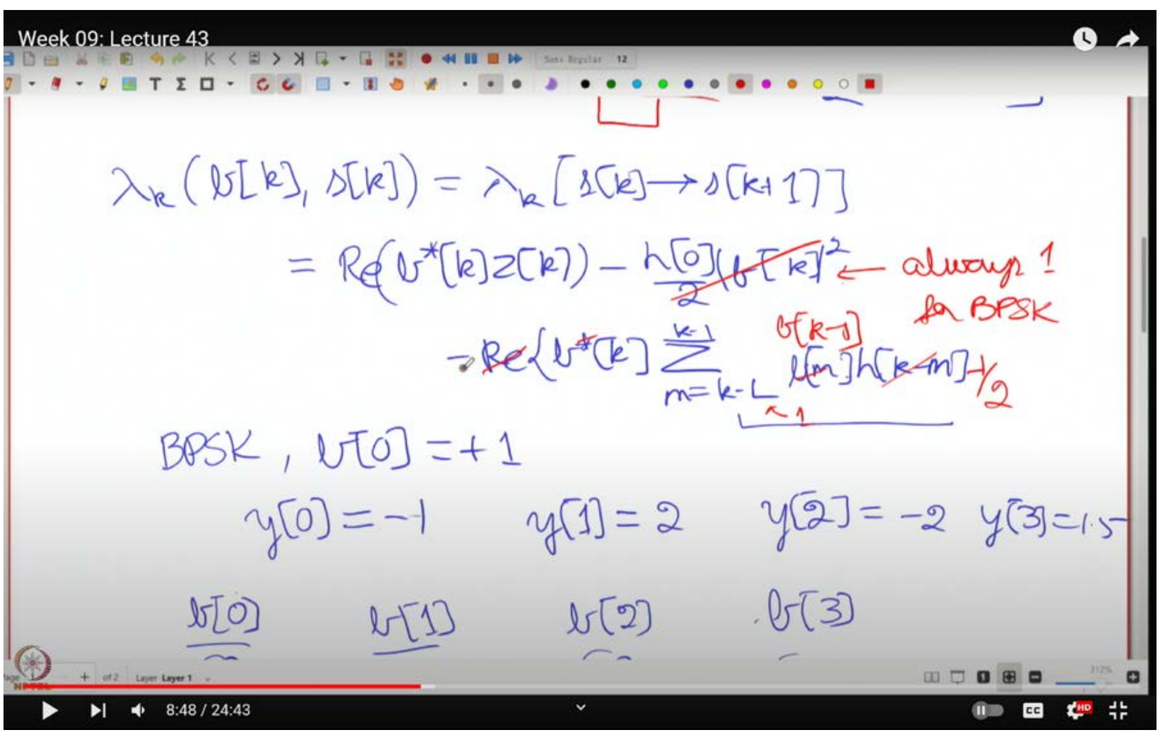
Now, for H_1, we must be more careful. To compute H_1, we use the integral of p(t) and p(t

- T), where T is the signaling interval. In our case, the signaling rate is T = 2, meaning we have half a symbol per second. To perform this integral, we shift p(t) by T = 2. Thus, the values will now be shifted accordingly: 1 is shifted to the position of $-\frac{1}{2}, \frac{1}{2}$ is shifted, and so on.

Once we perform this overlap integral, it becomes clear that the 1 from p(t) multiplies with the $-\frac{1}{2}$ from the shifted version, giving $-\frac{1}{2}$. This is the value of H_1, and it also happens to be equal to H_{-1}, since H_{-1} is the conjugate of H_1. Interestingly, in this real-valued case, there isn't much else to calculate for the higher H-terms beyond this point.

Now, moving forward, let us consider the incremental metric λ_k, which relates to the transition from S_k to S_{k+1}. We are going to use this incremental metric in our Viterbi algorithm. To make it easier to work with, let's organize our calculations and make a copy of this formula for further use.

(Refer Slide Time: 08:48)

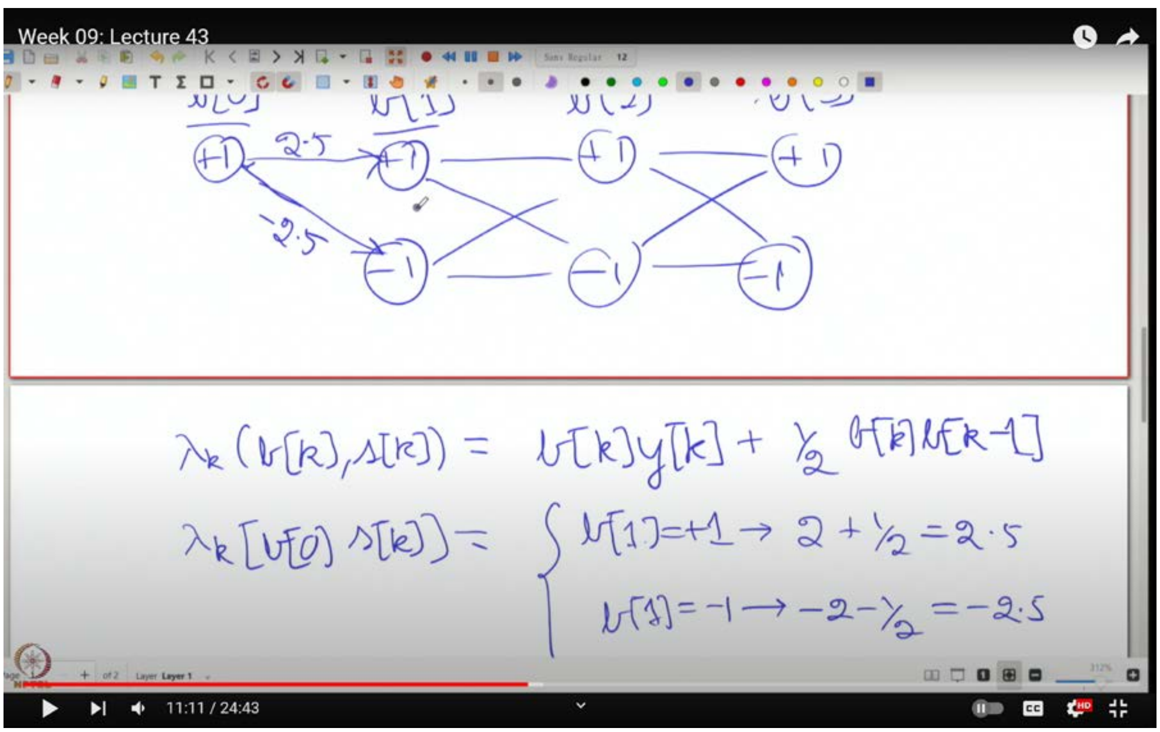


In this section, we will perform an example calculation using the Viterbi algorithm, specifically with Binary Phase-Shift Keying (BPSK) modulation.

First, let's define our setup. We will assume that b_0 is always +1. The sequence of received signals y is as follows: $y_0 = -1$, $y_1 = 2$, $y_2 = -2$, and $y_3 = 1.5$. These are the received values. Now, y_0 doesn't really affect the outcome since we already know that $b_0 = +1$. Therefore, we'll use y_1, y_2, and y_3 to determine what sequence was transmitted.

To proceed, we draw a trellis diagram representing the states at each time step. The trellis helps us visualize the possible values the sequence b_k can take. For b_0, the only possible value is +1. Moving to b_1, the possible values are +1 or -1. Continuing this, at each subsequent step, the sequence can take two possible values, either +1 or -1. This is essentially the interpretation of the incremental metric, which calculates the transition from one state S_k to the next state S_{k+1}, using the values of b_k and b_{k+1}.

(Refer Slide Time: 11:11)



Next, we need to compute the metrics associated with each branch of the trellis. These metrics help us evaluate the likelihood of different state transitions. We'll use the expression for the incremental metric that we derived earlier. However, before we do that, let’s simplify the expression based on some observations.

First, in BPSK, $|b_k|^2$ is always 1, regardless of the symbol. This observation holds for most QAM constellations as well. Moreover, h_0 is a fixed number, so we can disregard this term. Also, in our case, the value of L (the memory length) is 1 because our autocorrelation sequence depends only on L = 1. The autocorrelation sequence was $\{-\frac{1}{2}, 1, -\frac{1}{2}\}$, which also simplifies the computation.

Thus, the incremental metric simplifies significantly. It reduces to:

$$\lambda_k = b_k y_k + \frac{1}{2} b_k b_{k-1}$$

This expression is simplified further as we continue. Now, let’s plug in the values of y_k. We received the sequence y_0 = -1, y_1 = 2, y_2 = -2, and y_3 = 1.5.

We can now calculate the metric λ_k for each branch. Let’s start with the case where b_1 = +1. The metric in this case becomes:

$$\lambda_k = b_1 \cdot y_1 + \frac{1}{2} \cdot b_1 \cdot b_0$$

Substituting the values, we have b_1 = +1 and y_1 = 2, so:

$$\lambda_k = 1 \cdot 2 + \frac{1}{2} \cdot 1 \cdot 1 = 2 + \frac{1}{2} = 2.5$$

Next, let’s calculate the metric for the case where b_1 = -1. In this case:

$$\lambda_k = -1 \cdot y_1 + \frac{1}{2} \cdot (-1) \cdot b_0$$

Substituting the values, we get:

$$\lambda_k = -1 \cdot 2 + \frac{1}{2} \cdot (-1) \cdot 1 = -2 - \frac{1}{2} = -2.5$$

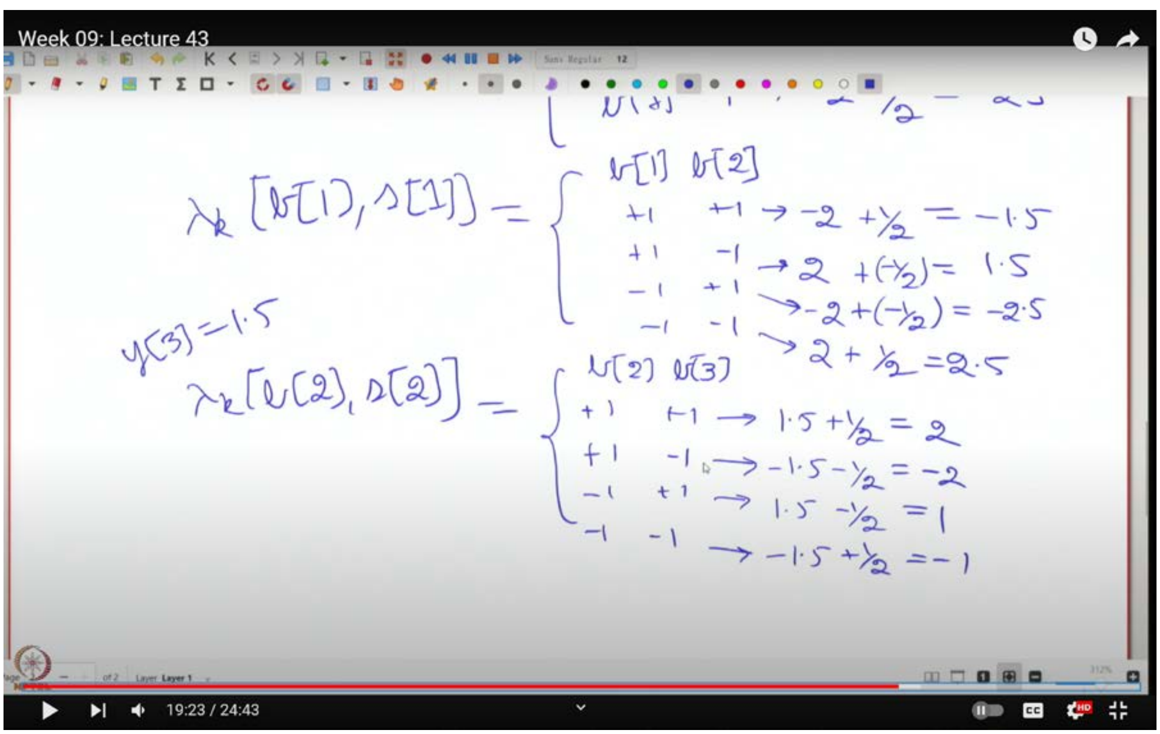
We’ll continue this process for the subsequent branches in the trellis diagram. Let’s calculate the metrics for the next transition, from b_1 to b_2, and so on. After calculating the branch metrics, we can write them on the corresponding paths in the trellis diagram.

For instance, the branch metric for transitioning from $b_1 = +1$ to $b_2 = +1$ is 2.5. Similarly, the branch metric for transitioning from $b_1 = -1$ to $b_2 = -1$ is -2.5. These metrics allow us to determine which path has the higher likelihood, meaning that we'll choose the path with the larger metric at each step.

However, note that you cannot make a final decision based solely on these incremental metrics at this point. Since the Viterbi algorithm relies on sequence-wise detection, you must evaluate the metrics at each step before making the final decision. This requires progressing through the entire trellis diagram, evaluating all possible sequences, and then selecting the path with the maximum overall likelihood.

Let's now proceed to the next stage in our calculation to further refine our decisions and complete the sequence estimation.

(Refer Slide Time: 19:23)



Let's analyze the different transitions for the case where b_1 can be either +1 or -1, and evaluate the corresponding metrics.

Recall from our discussion that each symbol depends only on the previous symbol. This relationship is straightforward in our case, as each transition depends only on the immediately preceding symbol. Specifically, b_1 depends on b_0, and b_2 depends on b_1. Now, let's compute the metrics λ_k for the various possible transitions.

Assume S_k is S_2. We need to compute the metrics for different combinations of b_1 and b_2. Given that y_2 = -2, we will evaluate the following cases:

1. Case 1: b_1 = +1, b_2 = +1

Metric calculation:

$$\lambda_k = b_2 y_2 + \frac{1}{2} b_2 b_1$$

Substituting b_2 = +1, y_2 = -2, and b_1 = +1:

$$\lambda_k = 1 \cdot (-2) + \frac{1}{2} \cdot 1 \cdot 1 = -2 + \frac{1}{2} = -1.5$$

2. Case 2: b_1 = +1, b_2 = -1

Metric calculation:

$$\lambda_k = b_2 y_2 + \frac{1}{2} b_2 b_1$$

Substituting b_2 = -1, y_2 = -2, and b_1 = +1:

$$\lambda_k = (-1) \cdot (-2) + \frac{1}{2} \cdot (-1) \cdot 1 = 2 - \frac{1}{2} = 1.5$$

3. Case 3: b_1 = -1, b_2 = +1

Metric calculation:

$$\lambda_k = b_2 y_2 + \frac{1}{2} b_2 b_1$$

Substituting $b_2 = +1$, $y_2 = -2$, and $b_1 = -1$:

$$\lambda_k = 1 \cdot (-2) + \frac{1}{2} \cdot 1 \cdot (-1) = -2 - \frac{1}{2} = -2.5$$

4. Case 4: $b_1 = -1$, $b_2 = -1$

Metric calculation:

$$\lambda_k = b_2 y_2 + \frac{1}{2} b_2 b_1$$

Substituting $b_2 = -1$, $y_2 = -2$, and $b_1 = -1$:

$$\lambda_k = (-1) \cdot (-2) + \frac{1}{2} \cdot (-1) \cdot (-1) = 2 + \frac{1}{2} = 2.5$$

These four metrics are then entered into the trellis diagram. Here's how they map to the transitions:

- For the transition from $b_1 = +1$ to $b_2 = +1$, the metric is -1.5.
- For the transition from $b_1 = +1$ to $b_2 = -1$, the metric is 1.5.
- For the transition from $b_1 = -1$ to $b_2 = +1$, the metric is -2.5.
- For the transition from $b_1 = -1$ to $b_2 = -1$, the metric is 2.5.

Thus, we will record these metrics in the trellis diagram, providing a clearer view of the likely paths based on the received sequence.

Similarly, the transition from +1 to -1 corresponds to a metric of 1.5, and the transition from -1 to +1 corresponds to a metric of -2.5. Finally, the transition from -1 to -1 corresponds to a metric of 2.5.

Let's pause for a moment to analyze these results. As we demonstrated in the previous lecture, these increments are additive. This means we can combine the metrics along different paths to determine the overall metric for each path.

Consider the case where we have two paths converging:

- Path 1: +1 → +1 → +1

The metric for this path is computed as:

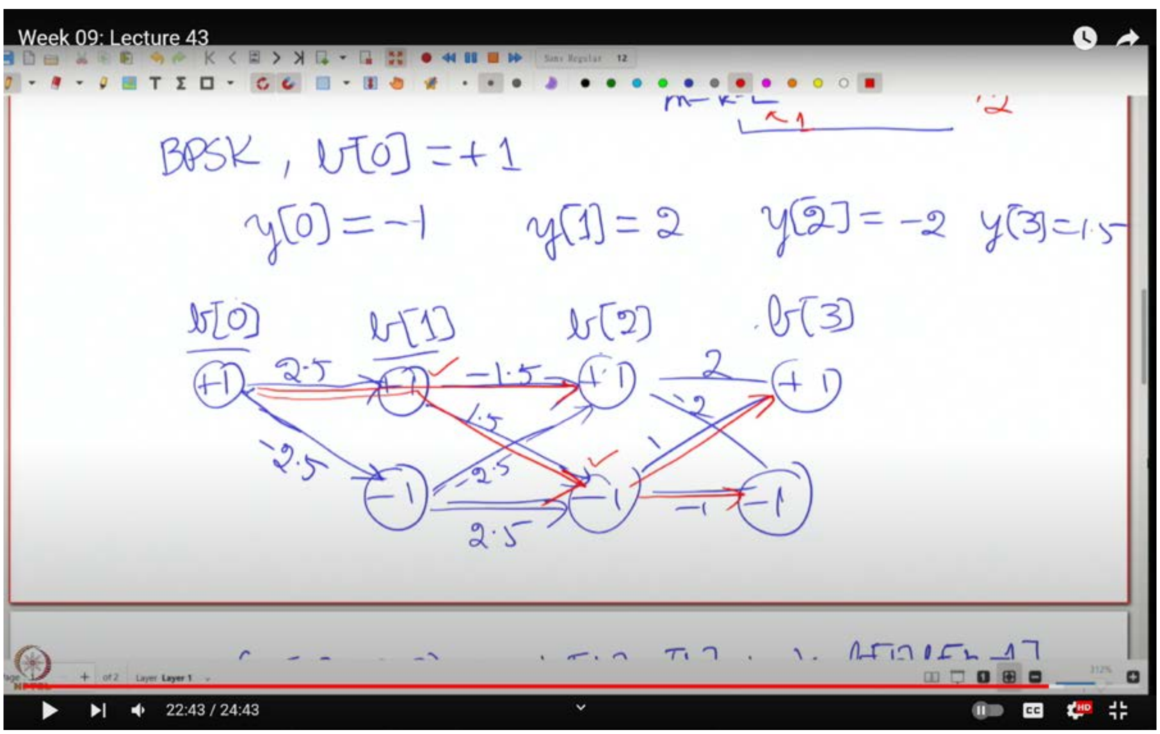
2.5 - 1.5 = 1.0

- Path 2: +1 → -1 → +1

The metric for this path is:

-2.5 - (-2.5) = -5.0

(Refer Slide Time: 22:43)



Since both paths converge at the same point, you can make a decision based on the metrics calculated. For the path +1 → +1 → +1, which has a metric of 1.0, it is considered optimal compared to the path +1 → -1 → +1, which has a metric of -5.0.

However, note that at the point where the two paths merge, there are additional paths merging from the -1 state as well:

- Path 3: +1 → +1 → -1

The metric here is:

2.5 + 1.5 = 4.0

- Path 4: +1 → -1 → -1

The metric is:

-2.5 + 2.5 = 0

At this juncture, Path 3 with a metric of 4.0 is clearly better compared to Path 4 with a metric of 0. However, a final decision cannot be made just yet because future symbols b_2 and b_3 will also influence the optimal path selection.

Hence, at this point in the trellis, we have identified the two most promising paths to continue with. These paths are:

1. +1 → +1 → +1

2. +1 → -1 → -1

These two paths will be retained for further consideration until we receive more information from subsequent symbols. Decisions can be refined as more metrics are calculated and new paths are evaluated.

Now, let's proceed to compute the metrics for the next symbol, Y_3 = 1.5, using the given formula:

$$\lambda_k = b_k y_k + \frac{1}{2} b_k b_{k-1}$$

For each possible transition, we compute:

1. Transition: +1 → +1 → +1

$$\lambda_k = 1.5 + \frac{1}{2} \times 1 = 2.0$$

2. Transition: +1 → -1 → +1

$$\lambda_k = -1.5 - \frac{1}{2} = -2.0$$

3. Transition: -1 → +1 → +1

$$\lambda_k = 1.5 - \frac{1}{2} = 1.0$$

4. Transition: -1 → -1 → +1

$$\lambda_k = -1.5 + \frac{1}{2} = -1.0$$

So, the computed metrics are:

- For +1 → +1 → +1, the metric is 2.0.
- For +1 → -1 → +1, the metric is -2.0.
- For -1 → +1 → +1, the metric is 1.0.
- For -1 → -1 → +1, the metric is -1.0.

Now we need to compare these metrics with previous calculations to decide on the optimal paths moving forward.

At this point, we have two surviving paths. To analyze them, let's consider the metrics along each path.

For the straight path from +1 to +1:

- The metric is calculated as follows: 2.5 - 1.5 = 1, then adding the next metric of 2, we get a total of 3.

Next, consider the alternative path that merges here:

- For this path, the metric is 2.5 + 1.5 = 4, then adding the next metric of 1, we get a total of 5.

Therefore, the path with the metric of 5 is the optimal one, and the path with the metric of 3 is discarded.

However, we also need to consider one more path:

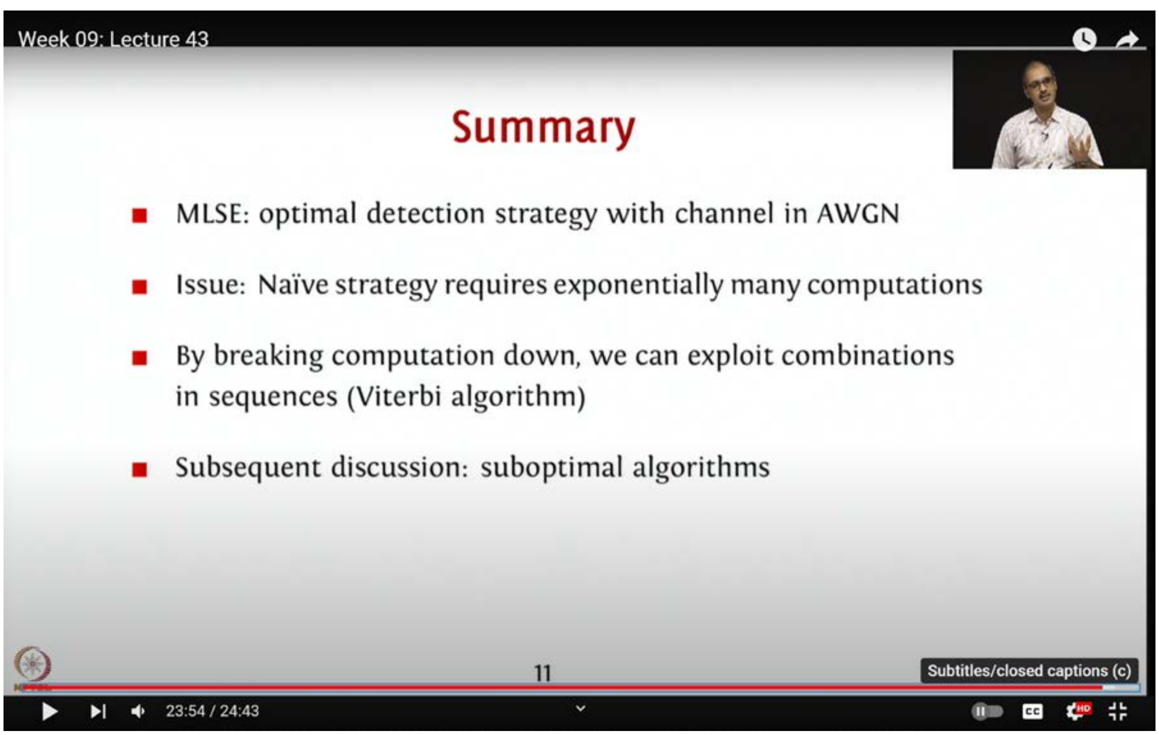
- The metric for this path is 2.5 + 1.5 - 1 = 3.

Thus, the two surviving paths are:

1. The path with a total metric of 5.

2. The path with a total metric of 3.

(Refer Slide Time: 23:54)



This decision is justified because, regardless of how far you continue, the metrics at the points where paths merge will not change. Decisions about past symbols are based on the available information without needing to revisit those past metrics. Therefore, when the optimal paths converge, you can confidently make decisions for symbols like b_2 and b_1. Even when paths don't immediately merge, you can still make optimal decisions based on the current metrics and the available information.

This method is the basis of the celebrated Viterbi algorithm, which is widely used for maximum likelihood sequence estimation. It is also applicable in Hidden Markov Model (HMM) scenarios, where continuous streams of input data are processed to compute probabilistic metrics and optimize decisions. The Viterbi algorithm can be implemented efficiently in Python, with numerous resources and libraries available for its application. For example, you can easily find Python implementations of the Viterbi algorithm by searching online.

In this course, we won't be implementing the Viterbi algorithm directly in GNU Radio due to its complexity and the cumbersome nature of such an implementation. Instead, we'll consider an alternative approach: while maximum likelihood sequence estimation (MLSE) is indeed optimal, it may be too complex for practical implementation in some cases.

So, what are the alternatives? Even though MLSE is theoretically the best, it might be impractical for certain applications due to its computational demands. In such scenarios, simpler, albeit sub-optimal, methods may be more feasible. These methods might involve simpler tasks like deconvolving the channel or identifying symbols through alternative approaches that don't achieve the same level of optimality as MLSE but are more manageable in practice.

To summarize our discussion so far: we've covered the branch metric computations necessary for MLSE, which is the optimal detection strategy in additive white Gaussian noise (AWGN) environments. However, a naive approach, such as trying all possible symbol combinations, requires an astronomical number of computations. For instance, for 1000 QPSK symbols, this could amount to 4^{1000} computations, which is impractically high.

By breaking down the problem and leveraging incremental metrics, we can simplify the process. This allows us to make decisions dynamically as we observe patterns and paths that suggest optimal decisions.

In our upcoming lectures, we will explore whether there are simpler, more implementable methods for decision-making in symbol detection. While the Viterbi algorithm remains optimal, practical constraints may lead us to consider other sub-optimal but more manageable algorithms. Stay tuned as we delve into these alternatives in the next sessions. Thank you.