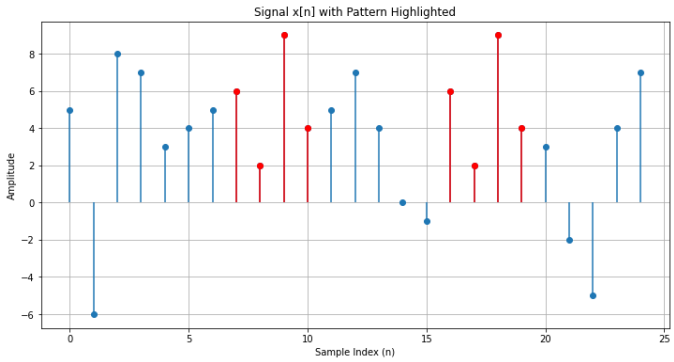
Signal Processing Algorithms & Architecture
Dr. Anirban Dasgupta
Department of Electronics & Electrical Engineering
Indian Institute of Technology Guwahati
Lec 6: Pattern Matching

Hello all, welcome to a new lecture on pattern matching. This is Dr. Anirban Dasgupta. And this is a course on Signal processing algorithms and architectures.

So, to start with, what is the problem of pattern matching in a signal? The word "pattern matching" usually comes with the concern of machine learning and artificial intelligence. So, in signals nowadays, people are trying to fit machine learning models but still classical pattern matching techniques are in use.



And they often outperform many machine learning models. So, what is the concept of pattern matching? So, the problem involves two steps. First, detect whether a specific pattern exists and second is if present, what is the time duration of that pattern? Like, if you see this figure above, so this is the signal marked with blue lines. And there is a repeating pattern marked in red, which has a duration of 4 samples and this pattern is typical if you see. 2, 9, and 4. Again, 6, 2, 9, and 4. So this is just for illustration purposes. That this looks so perfect, but this pattern may not be having exactly the same amplitude values. Also, the duration may not be exactly 4 samples. So let us go into detail about this problem. So why do we need to do pattern matching? And in signal processing, one of the main application is to identify some events which comes in the form of a pattern in the signal. And this event can also be some anomaly. So, for example, there is a waveform. This may indicate some events. Like in a heartbeat, you have to detect the heartbeat; there is a a specific pattern in the ECG signal.

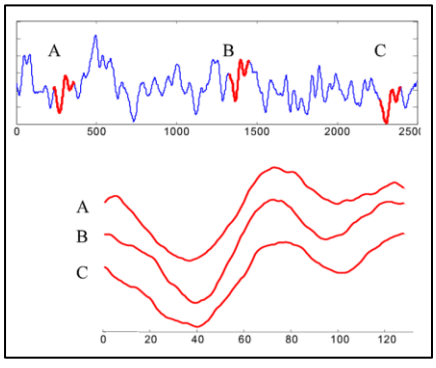
So, to count the number of heartbeats, you have to find exactly the patterns of this ECG QRS signal. So, as i said, in many biomedical signals like ECG and EEG. Specific

patterns appear, and these can be useful in diagnosing many conditions. In speech recognition, patterns can identify many things like phonemes words and maybe entire sentences. Then recognizing musical notes or instruments in audio signals, pattern matching can also help in denoising. For example, you know there is a saying this pattern is present in the signal, but the signal is noisy like this.

But if you know that this occurred pattern exists in the signal you can denoise it based on this pattern. Patterns can also help in signaling like compression, like if this is a pattern and if it is repeating, you can just store one pattern and the locations of the patterns, so that you need less memory to store the signal. So, what are the challenges in pattern matching? First of all, patterns can appear at random locations in the signal and then it is Not that once you have found the pattern, it is done. There can be multiple occurrences of the patterns.

And to make things worse, signals are usually noisy and makes pattern recognition more complicated and the pattern may be stressed or shrunk. In time, like say for example, heartbeat. So if you are capturing the ECG signal then the the pattern is similar, but each heartbeat is not of exactly the same duration.

Maybe there will be some differences in milliseconds. So, in terms of discrete time signals there can be differences in terms of data points. So that is again a challenge for the pattern. What you are trying to find may be stretched or shrunk over time and the amplitude can also be different. Like the pattern is the same; this is the pattern. So, this can be shrunk over time as well as enlarged in amplitude. So, these variations make a pattern matching challenging. So here we will discuss the pattern matching with the concept of time series motifs. So, what is a motif? So, motif is a subsequence in time series that occur multiple times and this motif in the time series will have similar characteristics or behaviors. It may not be exactly the same as if you see this pattern and this pattern and this pattern they are not exactly the same. The different amplitudes, but if you view these patterns at a glance, you will have the same or similar structures. So, the problem of motif discovery is to find such repeated patterns in a time series data, and then you need some distance measure, like if this is my motif. So, this is my signal. So, these are my three motifs: A, B, and C. They are similar but not exactly the same. So, I have to find some distance measures. Such that if I am going with motif A, I should be able to discover motif B based on some similarity measures as well as motif C. So, first let us represent this time series discrete time signal $x[n]$ as the values.



x_1 , x_2, x_3 up to $x_n$given by-

$$x[n] = [x_1, x_2, x_3, \dots, x_N]$$

N is the total number of samples in the series.

Now we will define a subsequence. So, subsequence is a section or a window of the signal which has length l and why length l? Because this is the length to which we are going to match my motif with the subsequence of the signal.

Like, say this is my motif and say a signal is happening like this. So, this has a length l and I have to search for windows of length L. Typically, sliding and overlapping windows and see where the match is best with the motif and this I represent by the subsequence or signal $S_i[n]$ given by-

$$S_i[n] = [x[i], x[i+1], \dots, x[i+L-1]]$$

Typically $L \leq N$

where L is the starting index from where the subsequence will start and typically this is less than or equal to N. So now, if I want to compare the subsequence and we need this distance metric, as I said, and the most naive distance metric in this context. It is a Euclidean distance, or to frame it simply. I have L number of values in the motif; I have L values in the signal subsequence L. I would say this is the motif so, I will just compare the differences for each corresponding points. So, this is my i, and this is $i + L - 1$. So I will compare each value, and then I will take the squared differences. So, this is the difference and then I take the squared difference sum the squared difference and take the square root; this is the distance. And if this distance is less than some small value. Threshold epsilon, because it will not be exactly 0; there may be some small value. So, if it is very small, less than a threshold.

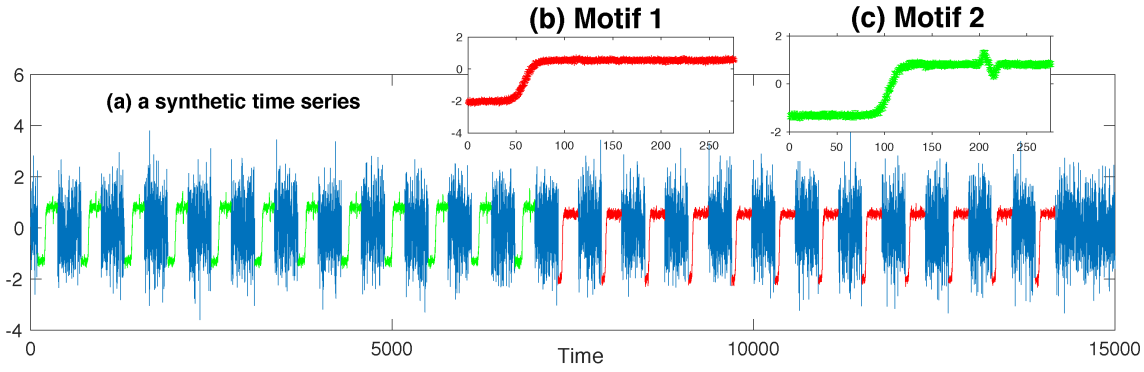
$$d(S_i, S_j) = \sqrt{\sum_{k=0}^{L-1} (x_{i+k} - x_{j+k})^2}$$

Then I would say that this subsequence. matches the motif or a pattern found at that specific location. So, the algorithm, as I explained, is first. We have to extract the subsequences and then we have to compute the pairwise distances, and then finally we will identify which subsequence matches the motif and whose distance is less than that predefined threshold epsilon. So, for matching a subsequence It will be $O(NL)$ because I have to perform L such comparisons or distance measures. So, if there are n sub-sequences considering that 99 percent overlap window, So it will be-

$$O(nL).$$

This is the complexity of motif searches and now to complicate things further. Say I do not have a single motif. I may have two motifs because if. This is a match; then this is red

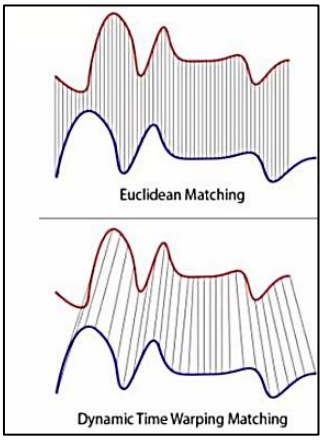
Motif, this is the match; this is a green motif. So, like this, if you have m number of such motifs,



Then the complexity will boil down to $O(NLM)$. Now this is a computationally expensive task. So, apart from the computational complexity, what are the other challenges? So, the challenges with Euclidean distance. That this Euclidean distance assumes that the sequences that are being compared are of the same length. But that may not be the case. Like I said, there are two hard bits. Hard bit patterns in ECG, but they are not exactly the same timewise.

Someone is taking n_1 number of samples. The other is taking n_2 number of samples where n_1 is not equal to n_2. So in that case, the Euclidean distance may not give you the best match.
Even if it loses the match, it will give the match is only the length of the motif. It cannot give more than that. So, this is the problem with this stretching and compression of the signals will not be accounted for.

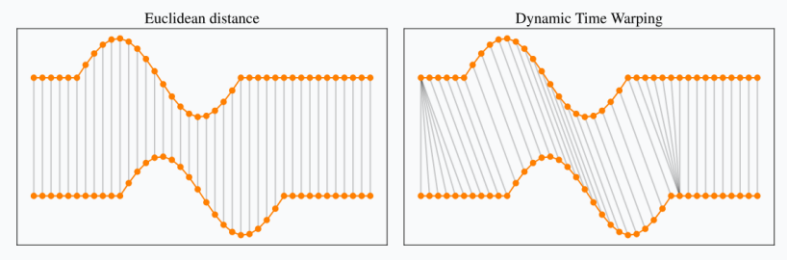
So, to solve this problem, there is one more technique which is called dynamic time warping (DTW), and this is a more flexible distance measure compared to Euclidean distance at which you can align two sequences which vary in length. What do you mean by varying lengths? So, these two sequences



have different elements, speeds, but they have the same pattern or a similar pattern or they can even if they are of the same length, they can be alike and are not synchronized temporally. So how does this dynamic time warping work? So, if you see that, there are two sequences. So, this green sequence and this blue sequence. Sequence, and they are very similar. But the problem is that the sequence is green, or sequence B is of length m while sequence A is of length n.

A is of length n, and m is not equal to n. So, the Euclidean distance in this case will Assume the match is for the same. Length of the motif sequence. But here you can warp the path such that you can match two sequences which are not of the same length. Like Euclidean matching will match exactly point-wise. So, this part is missed out whereas dynamic time warping adjusts the time points, the time lags accordingly.

So here you see

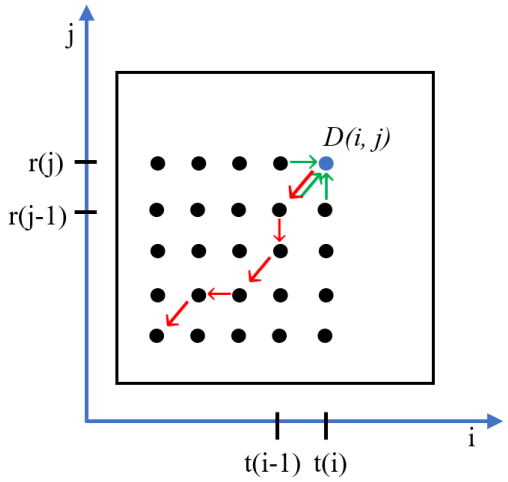


that there is a time lag. But this zigzag time lag is not accounted for in Euclidean matching. But DTW accounts for those time lags. So, this is to show the same thing clearly. So here, the matching is done one by one. So, this shift in the second wave is not accounted well for the Euclidean distance. Of course, if you use a sliding window approach then that will be accounted for but DTW can do it directly. So, here you see the pointwise matches are done appropriately. So, what DTW is doing is finding an optimal alignment and this alignment will be optimal in the sense that this will minimize the distance between the corresponding points, and this cost of warping. It is calculated by looking at all the possible subsequences.

So, this naturally looks computationally complex. Because you have to check for every crossing point as well. But do not worry, there is something in the algorithms called dynamic programming which makes this DTW computationally efficient to find the optimal path and then there are some other modifications which help DTW to find the solution in a faster way. So that is the main benefit that it will provide. For local shifts, even if the pattern is stressed or compressed, let us see this example. So, we have two sequences.

So, say t is the input sequence; this r is a reference sequence. Reference means our motif

in this case. So how will I find the DTW? So this DTW is appearing in the form of a matrix and this distance between two points is given by this distance matrix or this DTW.



t: input sequence
r: reference sequence

Local paths: 0-45-90 degrees

DTW of two sequences:

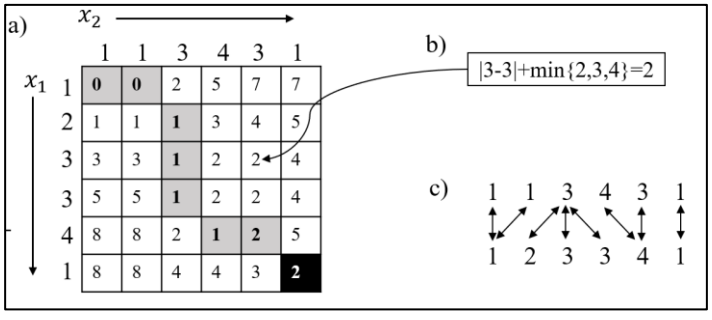
$$D(i, j) = ||t(i) - r(j)|| + \min \begin{cases} D(i, j-1) \\ D(i-1, j-1) \\ D(i-1, j) \end{cases}$$

Distance matrix by the formula the difference. between the consecutive points or the pair of points t(i-1) or r(j) where i and j may be equal not be equal to the minimum of these three values and this is the relationship that DTW is used in dynamic programming to solve this problem efficiently. If you are confused, I will give you one illustration that will make things easier. But what about the complexity? So, the complexity of n_1 and n_2 is for matching with one motive for one subsequence and then, if you have l subsequences, it will be. n_1, n_2, l, and then if there are m numbers of Motives, so this is a very high complexity.

As I mentioned in the beginning, but with dynamic. programming this gets reduced to n1 n2 and m. So how does it get reduced? So, if you see this matrix, now you see the formula looks pretty again complicated, but let us try to simplify this.

So, this is one subsequence, and this is one. Subsequently, and for simplicity, the sequence lengths are the same, but this is not any constraint, you can have different lengths as well. So, the first value is that I will check this.

So, this is the difference between 1 and 1. So what is the difference between 1 and 1? It is 0. So if I see the next value, then this is 1, and this is 1; the value is 0 and the minimum of the previous values, these.



So, what is the previous value? The previous value is also zero. So, this is 0 plus 0. What about this? So, 3-1 is 2 plus. What is the minimum value present? This is 0 on the left, just left of this and there is nothing above this.

So, 2+0 is 2. What about this value? So, this value will be 4-1, 4-1 + the minimum adjacent. So, there is only one adjacent which. It is 2, so 2. So, 4 minus 1 is 3, and 3 plus 2 is 5. What about this point? So, this point is 3 and 1. What is the difference between 3 and 1? The difference is 2. And then, what is the minimum? So, there is only one, which is five. So, 2 plus 5 is 7. You see that as we go. Away from the diagonal, the off diagonal, as it is going away from the diagonal, The distance is increasing, which is giving you the path of the lowest distance.

Now, if we say any arbitrary point, let's say this value, for example, so this value will be. We have to compare obtained by first the corresponding value, so 3 and 3, so 3 minus 3 takes the mod, so the distance is 0. And what are the three neighbors? This is one neighbor; this is one neighbor, and this is one neighbor. So out of these 3 neighbors, which is this the minimum? This is 2. Minimum is 2. So, this is 2. You take any value say this, so this will be 4- 3 is 1, and out of these which is the minimum? 1 or 3? So, it will take 1 plus 1, which is 2.

And now, once you complete the the whole matrix is now very easy. So first, we will start at this diagonal. And then we will reach this diagonal. And from here, what is the next step? So, I have three options: 0, 1, and 1.

So, 0 is the least. So, I go here. Then here I have three options: 1, 1, and 2. So among these three, both are equally less. So, you can take either one or the other. Now let us say I will take this. So even if you take this as well, this will not be a wrong solution.

So, if you see the next path, the possibilities are 1, 2, and 3. So naturally, 1 is lower. So I will take this path. So even if you take, instead of taking this path, if you take this path, that is also correct.

So, both answers are correct. Again from this 1, we have 2, 1, and 1. And 1 is lower, so I take this and similarly, I take this, then. I take this, and then I take this.

So this is the warping path. Now you see that 1 is mapped to 1. Then this 1 is again mapped to 1. Then 3 is mapped to 2. Again, 3 is mapped to 3 values.

That means the sequence in x_2 has 3 values in x_1. And then this 4 is mapped to this 4.

Then this 3 is also mapped to this 4. 1, and then the last 1 is mapped to 1. So, this is the alignment and the optimizations.

I would have used it if you see that I am not from this point, I am not considering all the possible paths because it has to move forward. So, that is called the monotonicity condition. That this will never go back in time index. Then the continuity condition that I am only going one step forward.

So, which is the continuity condition? I will not jump index then the boundary condition that I will start at this location and end at this location, and the last is the warping window. So, there will be a threshold, like if it is going away. I can set a threshold from only one direction of say 3 or 4 units such that if it is at the same level it up to 4 units; I will forcefully drop it down to the next location. So, this is all about pattern matching.

Thank you so much. Have a great day.