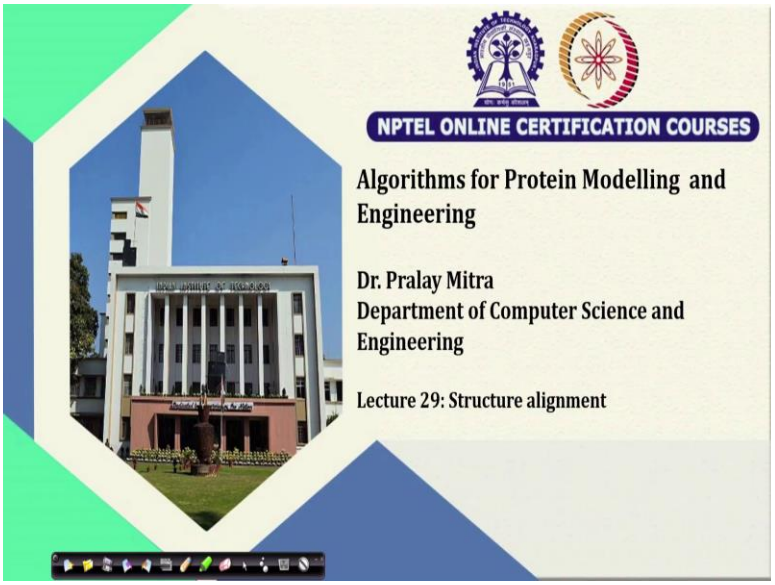
Algorithms for Protein and Modelling and Engineering
Professor Pralay Mitra
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture 29
Structure Alignment

(Refer Slide Time: 00:26)



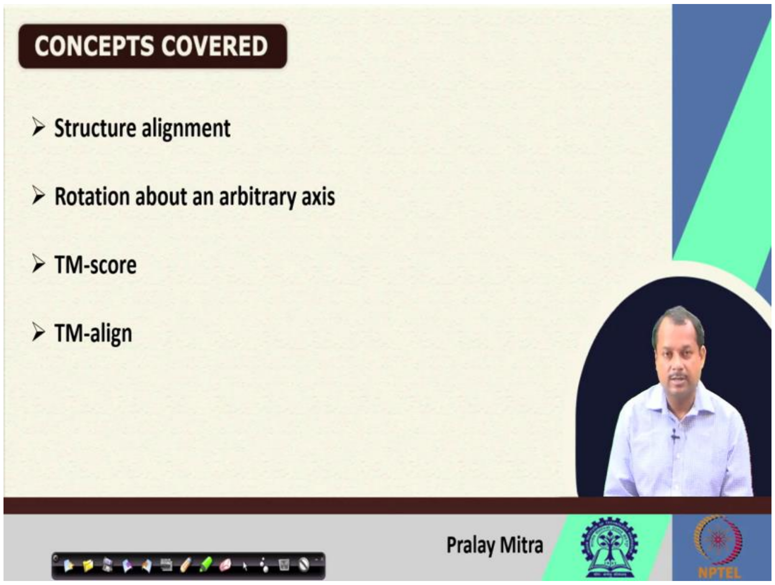
NPTEL ONLINE CERTIFICATION COURSES

Algorithms for Protein Modelling and Engineering

Dr. Pralay Mitra
Department of Computer Science and Engineering

Lecture 29: Structure alignment

(Refer Slide Time: 00:33)



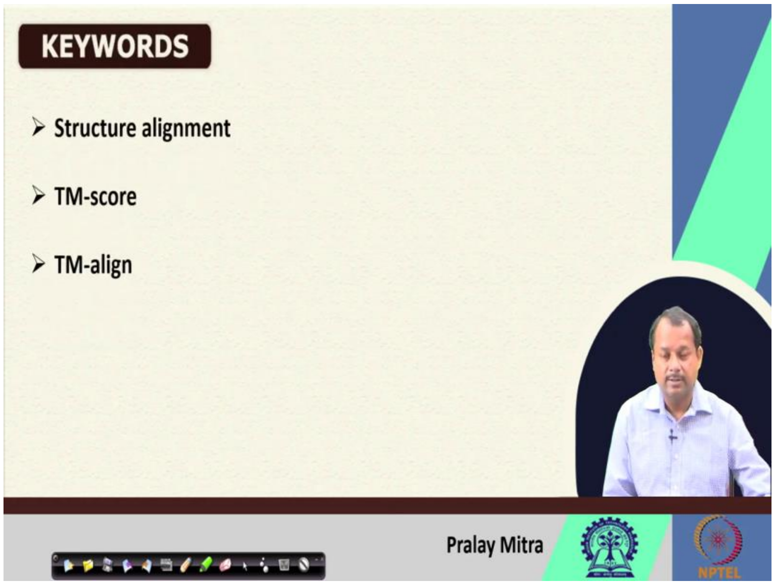
CONCEPTS COVERED

- Structure alignment
- Rotation about an arbitrary axis
- TM-score
- TM-align

Pralay Mitra

NPTEL

(Refer Slide Time: 01:23)



KEYWORDS

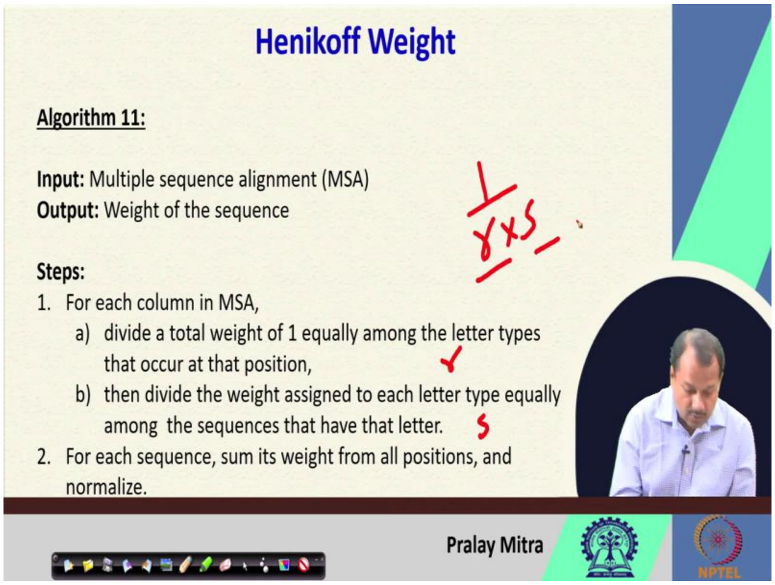
- Structure alignment
- TM-score
- TM-align

Welcome, back. So, in this lecture actually we will discuss about one structural alignment and before that one I was to explain the Henikoff weight that is very useful for going for this sequence with calculation say on the last lecture as I mentioned that when say I will go for when say I will go for aligning the sequences then multiple sequence alignment is done, now what you need to do that one new sequence has come you wish to put it there.

Now, when you are entering that new sequence what will be the weight of that new sequence? So, that information we use to calculate. So, there are different techniques and methods, so four different distinct techniques are there. Now, out of that one, I picked this Henikoff weight calculation for the discussion and because it is very fast, easy to calculate and also it is practically it is correct.

So, the concept here I am planning to cover that apart from the Henikoff weight, structural alignment, rotation about an arbitrary axis, TM score and TM align. So, accordingly these are my key words. Now, let us start with the Henikoff weight.

(Refer Slide Time: 01:29)



Henikoff Weight

Algorithm 11:

Input: Multiple sequence alignment (MSA)
Output: Weight of the sequence

Steps:

1. For each column in MSA,
    a) divide a total weight of 1 equally among the letter types that occur at that position, ✓
    b) then divide the weight assigned to each letter type equally among the sequences that have that letter. s
2. For each sequence, sum its weight from all positions, and normalize.

$\frac{1}{r \times s}$

Pralay Mitra

So, this is our 11th algorithm which says that input to multiple sequence alignment and output to weight of the sequence, that I need to calculate. So, the steps, so first for each column in MSA, MSA indicates multiple sequence alignment divide a total weight of 1 equally among the letter types that occur at that position.

So, we can consider that my total probability is 1 and then I am dividing that probability among the different occurrences of different letter type. Then divide the weight assigned to each letter type equally among the sequences that have that letter. So, that way if I consider that this is my r and this is my s, then it is very simple, it says that for each position, you just compute r multiplied with s.

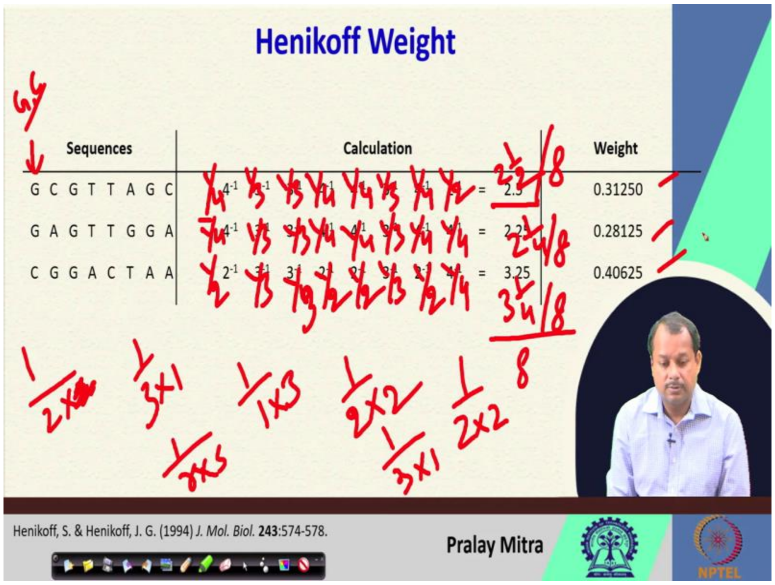
So, first at each position what I will do? I will look for the number of occurrences of different letters, and I will divide that information from 1, that is one division. Another division is that if one letter type occurs say a number of times, so I will divide its probability among those, say so say if I consider that in one position there are three different say letters occurring.

So, first for each letter the probability will be 1 3rd, and if one letter is occurring four times then the probability of one letter inside one string will be 1 4th of 1 3rd. So, 1 3rd is the probability for that particular letter, and 1 4th is a probability because four different letters are occurring, so that is why it will be 1 4th.

So, if I look at 1 3rd of 1 4th, then it will say 1 divided by 3 multiplied with 4 and that is what I wrote here. So, r multiplied with s. So, 1 indicates that in each column how many different data’s are occurring and another says that in each column, how many number of times one particular letter occurs. So, corresponding to that that that position, cell position, the probability of that cell position inside that multiple sequence alignment will be calculated.

Once you will have that one then for each such position you sum it up, you will get the total weight corresponding to one sequence. Now you calculate that what is the total weight contributed by all the sequences so that you can normalize their weight from 0 to 1, I mean that if you take the sum of them then you will get 1. So, that is my second step for each sequence sum its weight from all positions and normalize. So, let us give an example, then it will be very simple.

(Refer Slide Time: 04:33)



Henikoff Weight

Sequences	Calculation	Weight
G C G T T A G C	4^{-1} 3^{-1} 3^{-1} 4^{-1} 4^{-1} 3^{-1} 4^{-1} 2^{-1} = 2.5	0.31250
G A G T T G G A	4^{-1} 3^{-1} 3^{-1} 4^{-1} 4^{-1} 3^{-1} 4^{-1} 4^{-1} = 2.25	0.28125
C G G A C T A A	2^{-1} 3^{-1} 3^{-1} 2^{-1} 2^{-1} 3^{-1} 2^{-1} 4^{-1} = 3.25	0.40625

Henikoff, S. & Henikoff, J. G. (1994) *J. Mol. Biol.* **243**:574-578.

Pralay Mitra

So, here is one multiple sequence alignment on the left. So, this is a DNA sequences, so because you can see AC GT only is present. Now, it is not a problem, so what we can do for DNA sequence will be same for the protein sequences also. So, I am starting with this sequences and input is this sequence alignment. So, GCGTTAGC, GAGTTGGA, CGGACTAA, those are my multiple sequences alignment.

Now, I need to calculate they are normalized to it or incorporate, it is published in JMB long back in 1994. So, what I am doing in order to do that one, so first, so for this column I am considering, how many different, how many different characters are occurring? G and C. So, two different, so first I am dividing 1 divided by 2, now for this G how many times G is occurring? G is occurring twice again, so 1 by 2 again, so that is why 1 by 4, so this is 1 by 4.

So, here this 4 inverts actually this you can consider the nomenclature I have used 1 divided by 4 here. Now, this is also 1 by 4, this is C, now 1 by 2 multiplied with C is occurring only once in this particular column, so this part will not appear so this is 1 by 2. So, 1 by 4, 1 by 4, 1 by 2, now, CAG three different characters are there or sorry three different letters are there and character also you can consider.

Now, each is occurring only once, so all will be 1 3rd, 1 3rd, 1 3rd that I wrote. Next GGG, so only one character, so 1 divided by 1, but this GGG three different occurrences of G, so this is 1 3rd, again it will be 1 3rd, 1 3rd, 1 3rd. So, TTA, so 1 2 different characters, T is occurring twice, so it will be 1 4th, 1 4th and A is occurring only once, so half.

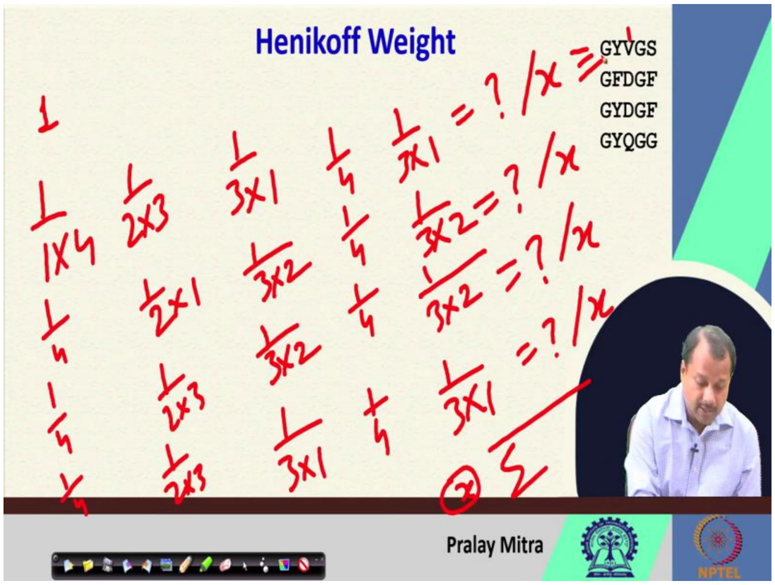
Then TTC, it will be same as the previous one only thing is that instead of A it is C that does not matter for us in this calculation. So, 1 4, 1 4, half. Then AGT, so three different, so 1 3rd and each is occurring once, so 1 3rd, 1 3rd, 1 3rd. Then GGA two different characters and G is occurring twice, so 1 4th, 1 4th, and A is occurring only once half. Then CAA, so it will be half 1 4th, 1 4th.

Now, if you add up along each column, you will see it is 1, and that is the thing we have done, so our proposal was 1 divided by r multiplied with s, which indicates number of times one character is occurring and how many different characters are there. So, we are considering these two. Along each row if I add it then I will get 2.5, so or 2 and half, then this is 2 and 1 by 4 this is 3 and 1 by 4.

If you sum it up then what you will get? So, 2 2 4 7 7 and this will give you 8. Now, if you divide this, divide by 8, divide by 8, divided by 8, then you will get this weight balance, very simple and very easy also to calculate, the algorithm also says that only thing what you have to do corresponding to each column you have to check how many different characters are occurring and how many times one character is occurring.

So, for that you can define one small structure, so where in each position it will have the information of to count basically interior values r and s and then it will compute and finally it will normalize based upon this one. So, this is the simple one.

(Refer Slide Time: 09:34)



Henikoff Weight

GYVGS
GFDGF
GYDGF
GYQGG

Pralay Mitra

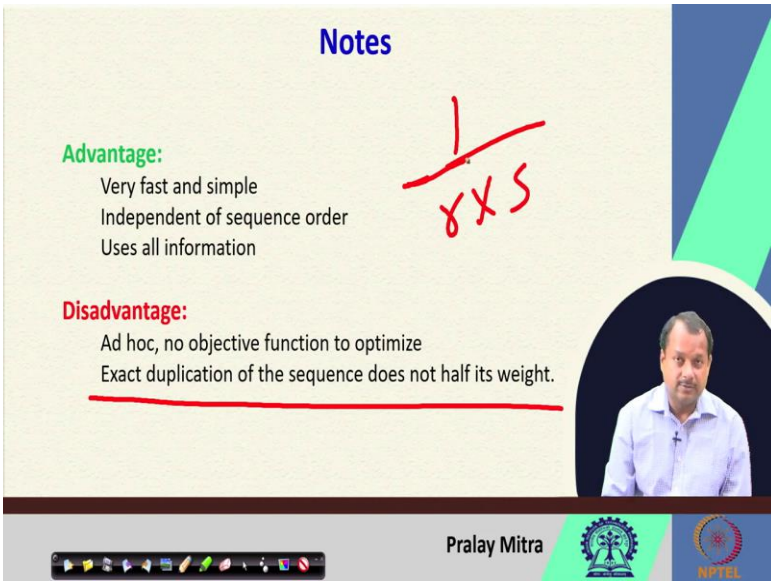
Now, if I give you another working example, say GYVGS GFDGF GYDGF GYQGG. If I assume that is my multiple sequence alignment, then how I can compute that one? Let us, start again it is very simple, so first column, so how many different characters are occurring? Only one, so it is 1 by 1, and in each position how many times how many G's are occurring? 4, so 1 by 4, if I consider for the first one.

For the next,1 by 4, 1 by 4, 1 by 4, YFYY, so two different characters are occurring and Y is occurring three times. So, two different characters Y and F and Y is occurring three times, so 1 divided by 2 cross 3, now for the F, so it will be half because F is occurring only once actually it is 1 cross 2, then for Y 1 2 cross 3, 1 2 cross 3.

For the next V, so VDDQ, three different characters are occurring, so 1 divided by 3, B is occurring only once fine, D is occurring twice, D is occurring twice, and Q is occurring only once. Then GGGG it will be same as the first one 1 4, 1 4, 1 4, 1 4, then SFFG three different characters are occurring, so 1 by 3, S is occurring only once, F is occurring twice and G is occurring, so this will be F also again 3 2 and G is occurring only once.

Now, if you take the sum then you will have some value, you calculate that one and then you take the sum for this, this value if I assume this is my X if I divide by X, then I will get a normalized weight value corresponding to that sequence. So, that is all about the Henikoff weight. I think it is very easy you can implement and you can calculate also given one multiple sequence alignment, you can tell me what will be the weight corresponding to each sequence.

(Refer Slide Time: 12:24)



Notes

Advantage:

- Very fast and simple
- Independent of sequence order
- Uses all information

$\frac{1}{r \times s}$

Disadvantage:

- Ad hoc, no objective function to optimize
- Exact duplication of the sequence does not half its weight.

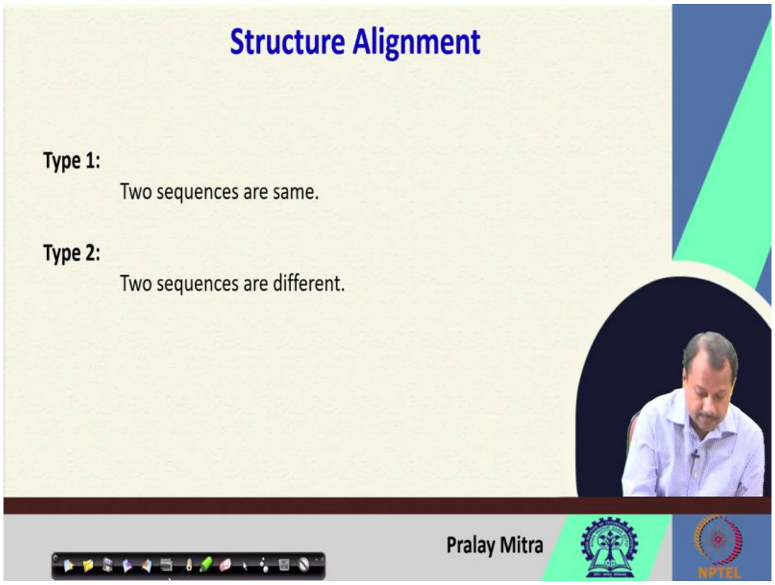
Pralay Mitra

As the note, so it is very fast and simple, it is independent of sequence order, it uses all information but disadvantage is it is ad hoc no objective function to optimize that you can understand from here and exact duplication of the sequence it does not half its weight. That is disadvantage sometimes you may consider that advantage also.

Now, here I am not explaining but I am keeping it open that why exact duplication of the sequence does not half its weight, you can think about it, and as a clue I can say that I mentioned say r 1 divided by r multiplied with s, where r and s indicates the number of times once one particular character is occurring and how many different characters are occurring.

Now, if you just duplicate it, I am not considering the terminal cases like there are only one sequence or two sequence or three sequences, let us assume there are at least ten sequences and then you are duplicating one sequence then that does not half its weight. Why? That you can justify.

(Refer Slide Time: 13:45)



Structure Alignment

Type 1:

Two sequences are same.

Type 2:

Two sequences are different.

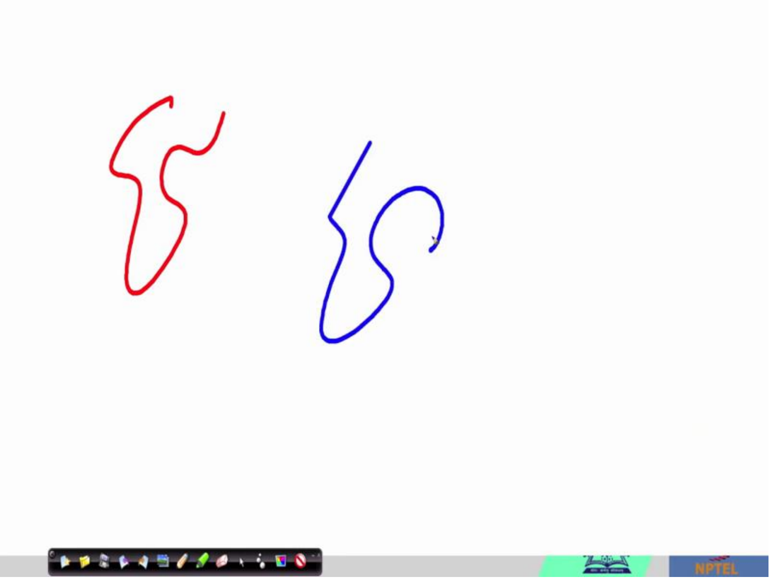
Now, regarding the structural alignment as of now probably it is clear to you that there can be of two type's situation, one is that two sequences are same. So, for that we did not, we did not have to go for the sequence level alignment, because if two sequences are same, then sequence level alignment is already done or known actually, directly you have to go for structural alignment. Otherwise, if sequences are different, you have to identify their correspondence and then you have to go for the structural alignment.

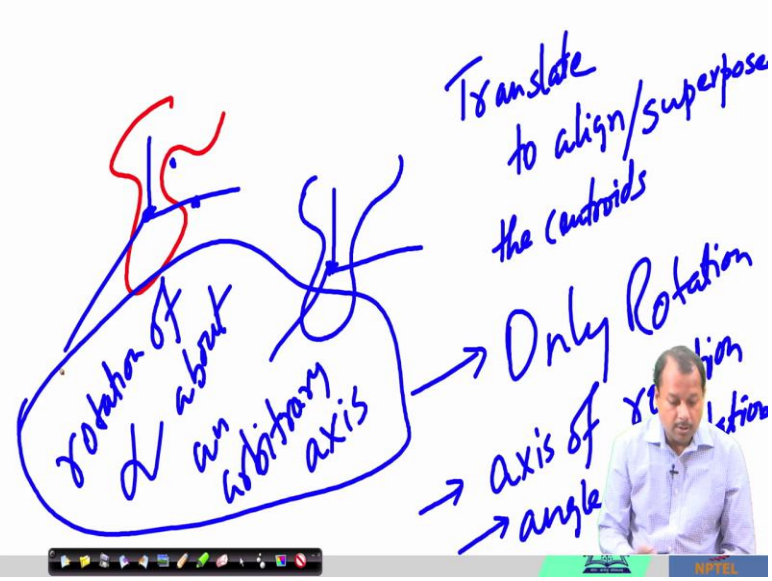
Now, if I assume that I have one case, it is the simplest one, then there are two say structures, same sequence is also same again, I can consider that the I got those two structures during the protein folding simulation, I mean one sequence was input my simulation was generating a number of different possible structures and I am interested to know whether the structure is actually same as the one say structure existing in a database or between two structures what is the deviation that I wish to compute.

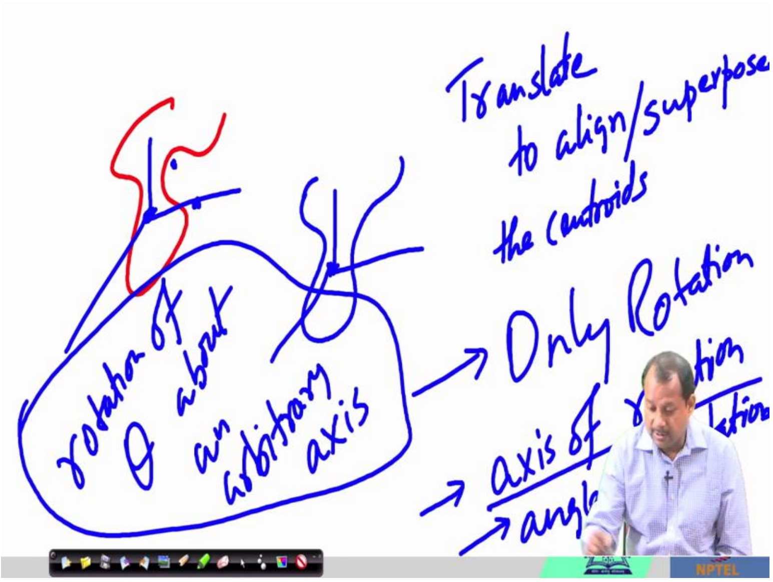
I already discussed one metric or measure to calculate their deviation that is root mean square deviation in short RMSD that we discussed and at the backbone level backbone trace level or all atom level those things we have discussed. Now, I am assuming that two sequences are same, because it is the same protein sequence different structures or decoys are given, I wish to calculate their alignment.

In order to do that one, so one thing you can do that two structures are given in order to identify their alignment, the best alignment what you can do that you take these two structures, you compute their centroid and translate these two structures, so that they are centroid with overlap with each other. So, after that translation process when their centroid will be overlapping with each other, then your problem will boil down to the situation, you have to give some rotation, so that they will be aligned properly. Now, regarding this rotation, what will be the rotation that is that question.

(Refer Slide Time: 16:24)







Now, for this my suggestion will be, so what you can do that for this what you can do basically, so one protein structure is say given here and say another protein structure is also given to you. Now, you wish to sorry this will not be actually both are same with minor difference, not this much of difference, because sequences are same, nevertheless if you consider that okay there will be much difference because I am doing protein folding problem then that is a different issue, I know, but something like this.

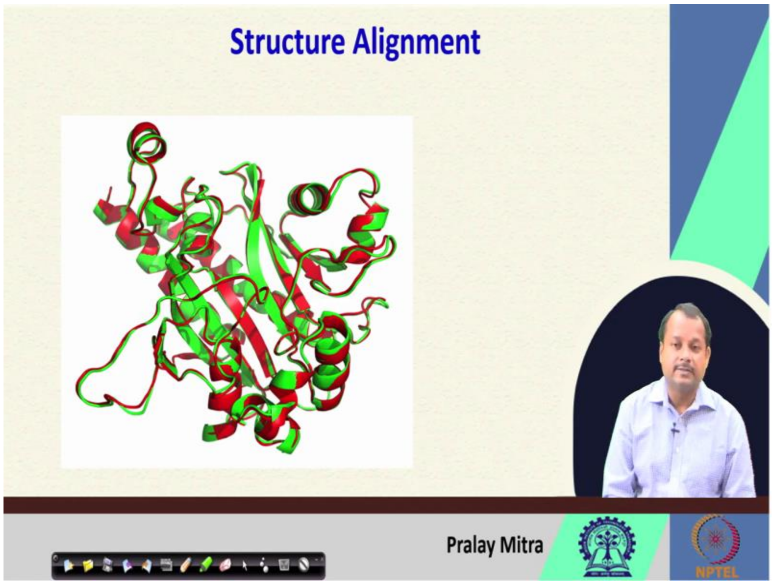
Now, what I am suggesting, so first you computed the centroid, so if I consider this is the centroid and this is the centroid then you give one translation. Next it will be only rotation. Now regarding this rotation, when it is then two things you need to know, axis of rotation and angle of rotation. Now, regarding this if I consider that say one reference frame is here and that is something like this, so in this reference frame how I can say define, so it is simple, so that you consider that origin of your reference frame is basically the centroid for this say red protein.

Now, you pick three non-collinear points, maybe three different amino acids as maybe three CL fat off three different amino acids, which are not collinear in nature, and then you have one along the x axis and allow another on the x y plane, and that way you can have one coordinate system. Similar to that you can have another coordinate system here, now with after the superposing the centroid, the centroid of these two-coordinate system will be same.

Now, the same three points here that you identified for the red and for the blue you compute that what is the angle, and what will be the axis corresponding to that one. Now, after knowing that one, so you need the axis and you need, you know the angle of rotation. Then your problem will boils down to rotation of alpha, assuming alpha is going to be your rotation about an arbitrary axis or say actually I will use alpha beta gamma, so better use some things say theta. Now, you know both, axis of rotation, angle of rotation. Then, then my suggestion will be that how you can perform this one.

So, this type two I will come later and then we will discuss TM align. So, first these two sequences are same, let us assume. So, second type two also we can use the same one which I discussed, but let us first discuss the simplest one that is two sequences are same, correspondence are known to you, you computed the centroid, you align them and then you identified that axis of rotation and the angle of rotation, now your job will be to rotate about that axis here I am calling rotation about an arbitrary axis to align two protein structures.

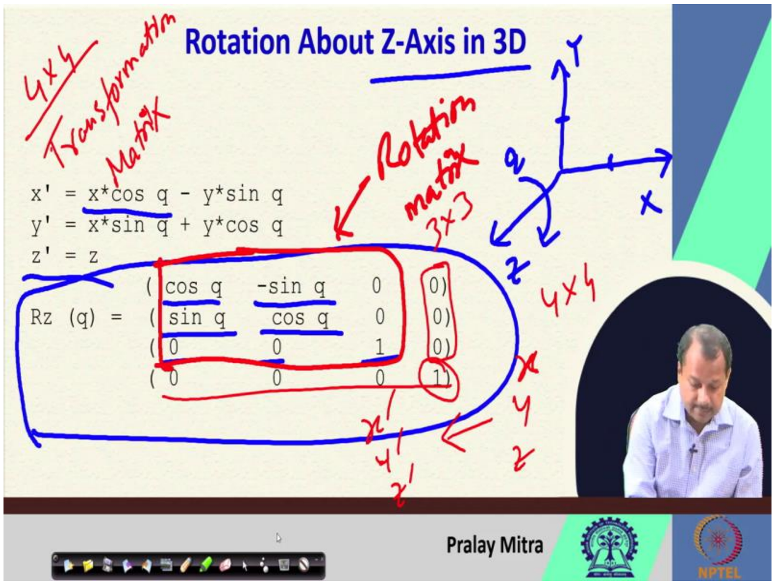
(Refer Slide Time: 20:54)



Structure Alignment

After that alignment something like this will come where green and red are two protein structures and from here you can see they are with the same structure probably their sequences are also same and yes, it is.

(Refer Slide Time: 21:08)



Rotation About Z-Axis in 3D

x' = x*cos q - y*sin q
y' = x*sin q + y*cos q
z' = z

$$Rz\ (q) = \begin{pmatrix} \cos q & -\sin q & 0 & 0 \\ \sin q & \cos q & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Pralay Mitra

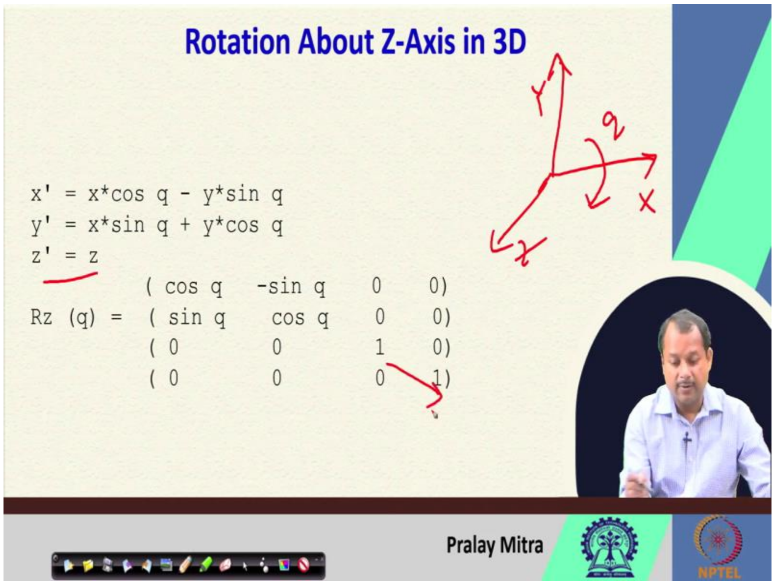
Now, to start with, so first if you go for rotation about z axis in 3D, then your rotation matrix will be looking like this. So, what is the rotation matrix? This is your rotation matrix. And how it has come? Since it is your rotation about z axis which means you are rotating about this and you are rotating say angle of say q, if it is then you can write after the rotation the coordinate x will be converted to x prime, y will be converted to y prime, and z will be converted to z prime.

But since you are rotating about the z axis, so the point z will be same as the z prime that is why z equals to z, sorry z prime equals to z. Now, if you rotate about this one, then there will be a projection of that angle on this x and on this y, and from that projection the equation that you will get x cos theta or in this case q, x cos q minus y sin q, and for y prime you will get x sin q plus y cos q. In this matrix form this can be written as cos q minus sin q, sin q cos q and for z it is 0 0 1.

Now, this one is actually your rotation matrix, if you, this is your rotation matrix. If you multiply these rotation matrix with x y z then you will get x prime, y prime, z prime, but to make it complete sometimes instead of this rotation matrix that you can see is a 3 cross 3 matrix can be written as 4 cross 4 where 4th row will indicate 0 0 0 1 and 4th column will be 0 0 0 0, this will corresponds to translation and this will correspond to scaling.

Since we are not considering any sharing or scaling of the object and we are keeping it fixed, so it will be 1 and already we translated that 1, that is why this will be 0 0 0. But if we incorporate translation also then this will be non-zero values. Basically these 3 cross 3 is the rotation matrix, but along with that one this these are appended in order to make it a 4 cross 4 matrix, and that 4 cross 4 is called as the transformation matrix. Because it includes rotation, it includes translation, it includes scaling, sharing et cetera, so it is basically transforming the matrix. So, that is about rotation about the z axis, in 3D, because x y z three coordinates are there.

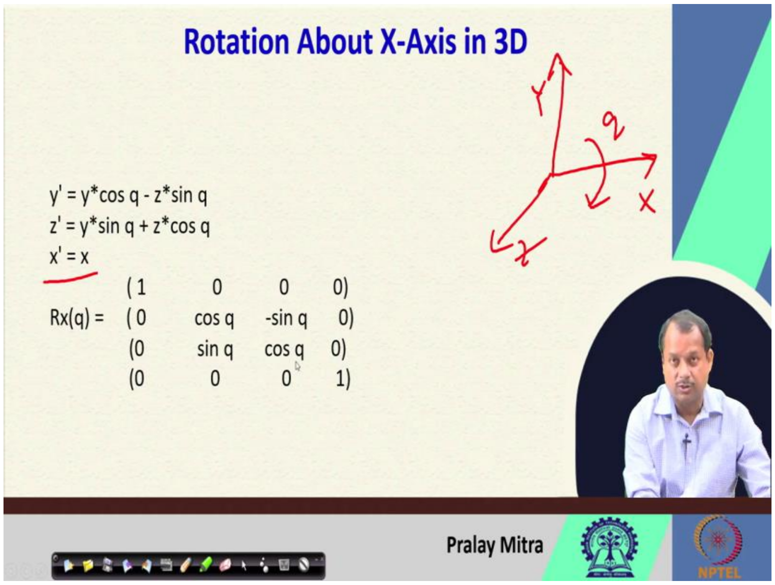
(Refer Slide Time: 25:01)



Rotation About Z-Axis in 3D

$$x' = x*\cos q - y*\sin q$$
$$y' = x*\sin q + y*\cos q$$
$$z' = z$$

$$Rz\ (q) = \begin{pmatrix} \cos q & -\sin q & 0 & 0 \\ \sin q & \cos q & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



Rotation About X-Axis in 3D

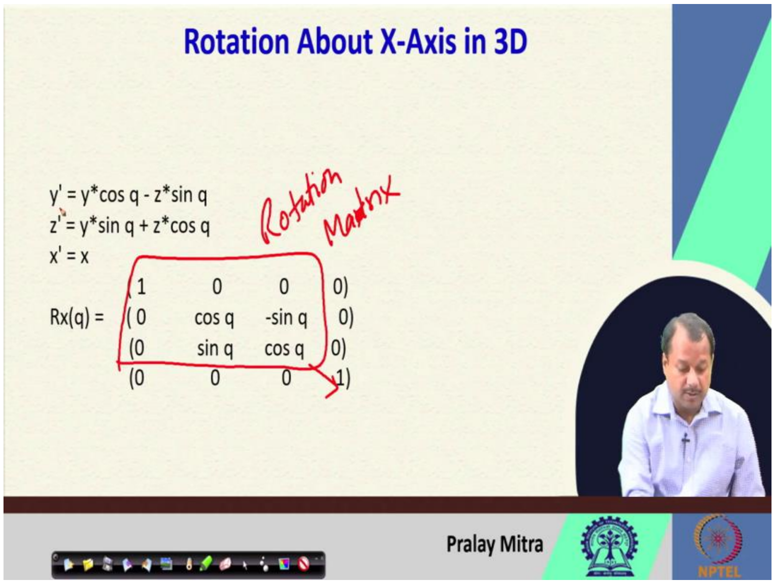
$$y' = y*\cos q - z*\sin q$$
$$z' = y*\sin q + z*\cos q$$
$$x' = x$$

$$Rx(q) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos q & -\sin q & 0 \\ 0 & \sin q & \cos q & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Now, if instead of z axes, I move on to say y axis then, what is happening you see sorry about x axis first, then since it is about x axis, so and again the angle I am keeping say q, so x prime will be x, there will be no change, others will change. Now, you notice the change, if I go back to the previous slide, so cos q minus sin q, sin q cos q, so, it is sliding.

So, in order to remember it correctly, so this cos q is coming here minus sin q, sin q cos q, and here it is 1, so you can consider that here there is one bit say here there is one bit say shifting and because of that shifting what is happening this 1 is going here, 1 0 0, this cos q minus sin q is coming here, so this way actually kind of shifting, that you can consider.

(Refer Slide Time: 26:16)



Rotation About X-Axis in 3D

y' = y*cos q - z*sin q
z' = y*sin q + z*cos q
x' = x

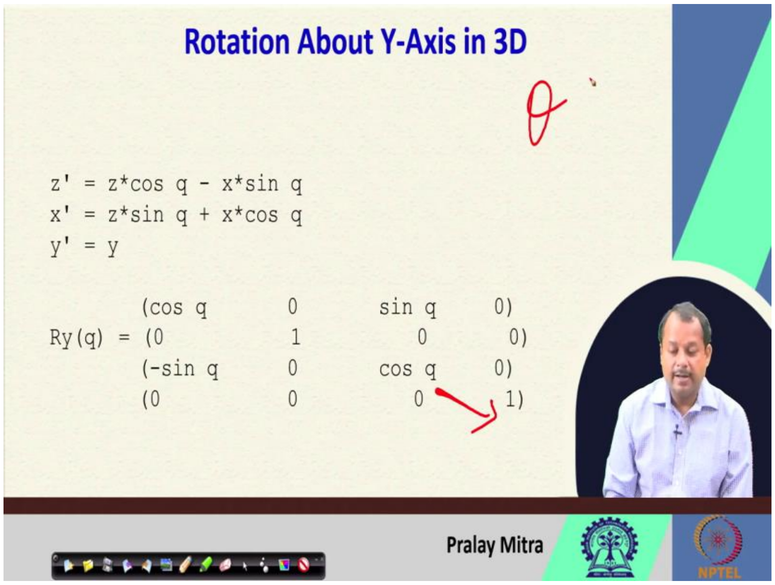
Rotation Matrix

$$Rx(q) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos q & -\sin q & 0 \\ 0 & \sin q & \cos q & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Pralay Mitra

So, diagonal shifting because of that one what will happen you see that this 1 has come here cos q minus sin q, sin q cos q. Now, this is our rotation matrix. Now, you know that rest of the part is 4 cross 4 that is the transformation matrix that I mentioned, so this is my rotation matrix about the z axis, x axis. Now, if you go for y then it will be another say translation, so rotation about the y axis, so definitely y prime is going to be the y, y prime is going to be y.

(Refer Slide Time: 26:56)



Rotation About Y-Axis in 3D

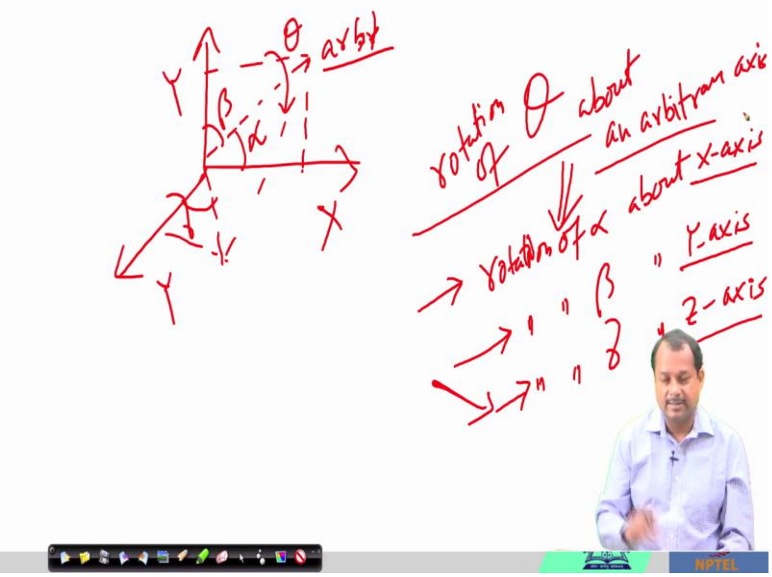
θ

```
z' = z*cos q - x*sin q
x' = z*sin q + x*cos q
y' = y

          (cos q      0      sin q     0)
Ry(q)  =  (0          1        0        0)
          (-sin q     0      cos q     0)
          (0          0        0       1)
```

Now, cos q 0, sin q 0, 1 0 minus sin q 0 cos q. So, one changes has done and because if you consider that it is the translation this way then it will be simple for you to remember. So, we are getting three different rotation about x y and z axis. Now, if I go back and tell you that, so given two protein structures, they are same sequence wise and we computed their centroid and we aligned the centroid after aligning their centroid, we identified three non-collinear points and based upon that one we identified the axis of rotation and the angle of rotation, then if I assume that axis of rotation is something and angle of rotation is say theta, and if I assume that this angle of rotation is theta.

(Refer Slide Time: 27:56)


Y
θ
arby
β
α
X
Y
rotation of θ about an arbitrary axis
rotation of α about x-axis
" " β " Y-axis
" " γ " Z-axis
NPTEL

Now, I have x y z coordinate system, if my angle is say sorry this is my theta, this is my theta and I need to rotate about this not this thing, this is not theta actually, if I need to rotate about this axis, if I need to rotate about this axis arbitrary, arbitrary axis, so this match is fine, if I need to rotate that one then what you can do that you can actually project this theta on to x, on to y and on to z, and that way here you will get alpha, then here you will get beta and here you will get basically gamma.

So, rotation of theta about an arbitrary axis translates to what rotation of alpha about x axis, rotation of beta about y axis, rotation of gamma about z axis. Now, I know what will be the rotation matrix for rotation about x axis, what is the rotation matrix for rotation about beta and what is the rotation about z axis.

So, x y z, each matrix I know, so what I can do? I can give rotation about x y or z or what I can do, I can have one 3 cross 3 matrix or 4 cross 4 transformation matrix where I multiplied rotation about x axis, rotation about y axis and rotation about z axis and I computed one resultant matrix. Then I combine that information, then I will have only one resultant matrix, not resultant actually the final matrix, and that matrix I will actually use that matrix I will use in order to transform the points from x y z to x prime y prime z prime which will actually give me the rotation about an arbitrary axis.

So, that is the thing, now in the next lecture we are going to sum up those things into different steps form, so that we can actually transform the or not transform in this case only the rotation or if I include the translation also along with the rotation then I can transform one protein molecule so that it will align with another protein molecule. So, that steps I will mention along with that one other TM align and TM score that we will discuss on the next class, next lecture. Thank you.