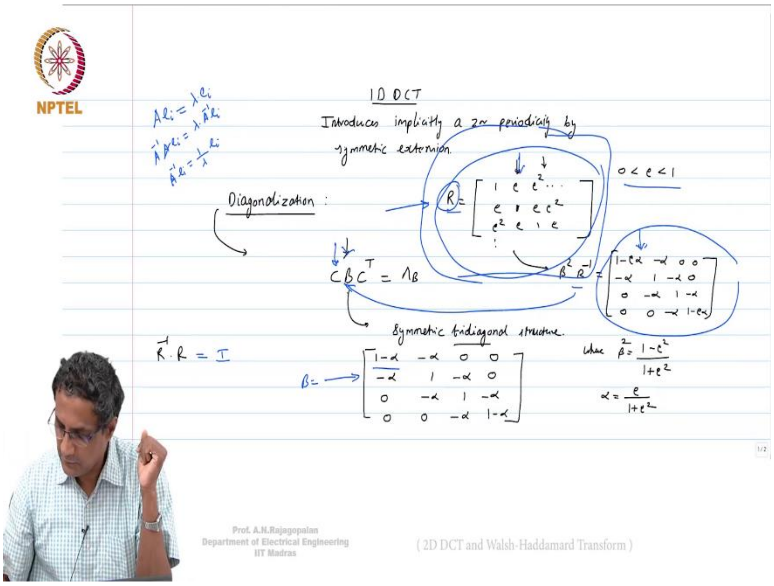
Image Signal Processing
Professor A.N Rajagopalan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture 50
2D DCT and Walsh-Haddamard Transform

(Refer Slide Time: 00:24)



So, yesterday, we just, we just about, talked about, yeah, I just showed you that you can actually compute DCT through a DFT. And it also means that, what it also means is that the DCT introduces something like a kind of, it introduces an implicit sort of a periodicity, introduces implicitly rather not explicit, but implicitly, introduce implicitly a 2N periodicity, 2N periodicity by symmetric extension, by symmetric extension of the sequence.

And we also saw that because of this, you do not have false jumps and therefore, it has a, it has a decay, decay rate that is actually higher than, higher than this one, DFT. Now, one more thing, like, like I said, if you look at the, look at the diagonalization property, this does not have, the first point, it does not have anything to do with diagonalization simply the fact that the when you take, when you take a DCT type 2, this is what it does. The other property is one of, one of the diagonalization. And this, and this diagonalization is again yet another sort of a sort of a property of DCT that actually propels it over DFT.++++

And this is a fairly important property because this applies this, this something like an, like an ansible property, this applies for a whole class of signals. So, right when you talk about, when you talk, when you talk about an (())(01:45) Markov process right, it looks like this 1 rho rho square, all the way and then it will say rho 1 rho square and then rho square, rho 1, rho and so on.

So, rho path N minus 1 whatever, at this end and then goes all the way down to 1. Now, this kind of a matrix, we know and where this rho, rho now takes a value between 0 and 1. Now, now the interesting part is, now if you ask him what kind of a matrix does this guy diagonalize, when you look at a 1D, 1D DCT, suppose, you asked it what kind of a matrix does it diagonalize?

It turns out that it diagonalizes, these are all works that others have done and they have shown that, just as we did, suppose you have some matrix, lets say, whatever, you want to call it, some A and, or lets say, lets call it some B, so that you do not confuse it with some, so B. Then what we did was for a, for a Fourier, we did 5B 5 star or typically to be 5 Hermitian mission. In this case, we will have to do like CBC transpose because C is real.

Now, when you do this, this is supposed to diagonalize, as we but then the point is what kind of B does, does this diagonalize. For a Fourier transform, we saw that B was circulant, the Fourier will actually diagonalize it. Now, with respect to the, with respect to the, with respect to a DCT, one would wonder as to what kind of a matrix it diagonalizes.

So, I will just write on the structure of the matrix that it diagonalizes, it is a very kind of a curious structure. And at first sight, it is not even clear, I mean why is that relevant? Okay, but let me write it down. So, this looks like suppose I write down for a 4 cross 4, I mean you can write it for the general case. So, this, so this matrix, this is, this has a symmetric tridiagonal structure, symmetric tridiagonal, tridiagonal structure, and it looks like this, 1 minus alpha minus alpha 00 minus alpha 1 minus alpha 0 0 minus alpha 1 minus alpha 0 0 minus alpha 1 minus alpha.

So it looks like this, so for example, if you take, if you take how, if you are going to construct a C which is 4 across 4, and suppose you multiply it in this manner, you will see that it actually diagonalize it and then those will be the eigenvectors. Eigenvectors will be the columns of C transpose and all you can see, just as we did for Fourier. I do not want to do all that now, you know how to do. Now, what has this got to do. So, the fact that, that it can diagonalize a matrix like this, it is not immediately apparent as to why is this so, why is this so important.

Now, it turns out that, that covariance like this, when you write down which is like, which is like a (())(14:53) and this is normally followed by most of your, most of your natural signals, right? So, it is like saying that, it is like saying that on the immediate neighbour, you are going to see sort of a dependence.

You know, in terms of rho, imagine that rho is, let us say, 0.8 0.9 or something as soon as you go further off, it reduces. For example, if you are sitting here with, with yourself, you are 1 and then with your neighbour on either side, it's like rho, and then as you go farther off, so it's like rho Yes?

Student: How does this matrix generalize to a larger size?

Professor: How do you…

Student: (())(05:24)

Professor: Yeah, so onto the right, because it sets a tridiagonal matrix, so we will get three 0s onto the right, so the diagonal will end and then you will get one more, 1 in the middle, you get like 3 1s in the middle 1 minus alpha 1 minus alpha and then all minus alphas and minuses. So, it will remain a tridiagonal except that one more 1 will get introduced in the middle, so same way it goes to whatever size.

Student: (())(05:48)

Professor: So, what does this got to do with what, what, why is this property important. Why is the fact that if a diagonal is a such a matrix, it is good? The point is when you look at R, which is a covariance like this, it turns out that, no, no is a, right this, this we are not trying to get off, we are not going to prove this but then one can show that beta square R inverse, beta square R inverse, it has this form.

This again I am going to write for a 4 cross 4. If you have the same 4 cross 4 then the beta square, R inverse takes up a form, which looks like well, which is, which will be what looks like 1 minus rho alpha minus alpha 0 0 and minus alpha 1 minus alpha 0, then 0 minus alpha 1 minus alpha, minus alpha, and then 0 0 minus alpha 1 minus rho alpha. Turns out and in fact that you can show this, if you are still doubted what is beta and alpha where beta is given by 1 minus rho square, beta square in fact is given by 1 minus rho square by 1 plus rho square and alpha is given us a rho by 1 plus rho square.

So, what this actually means is that if you try like for example, if you take R inverse to be 1 by this, this quantity, the beta square and suppose you do R inverse times R, where your R is this R, where your R is this that you can show that, if you take R inverse to be 1 by, 1 by C beta square, this matrix on the right hand side and if you do this right then you'll get actual identity. This I leave it to you to show.

Now, what is going to see throws up as this is? Interesting thing that for example, as rho, rho comes closer and closer to 1 in the sense that they are rho 8.8 5.9 as rho, as rho tendency tends to 1, then what happens is, what happens is like this matrix that you have here begins to look somewhat like this, though it is not exactly this. Correct, you have 1 minus alpha there, but then as rho tends to 1, the, the (())(8:11) structure of this matrix right that is sitting here looks, begins to look, begins to look, no, no, very much like this.

Now, so, so, actually, so, the idea is that when you when you hit something like rho 2.95 and so on right, so what you will find is that, is that DCT, suppose, suppose for, for this rho equal to 0.95, if you take right this to be Rn, we also know that, see, it is like this, right? So, if I if I know that, if I know that you know that this C diagonalizes R inverse because of the fact that because of the fact that, I mean when you know when this when this rho kind of right comes close to 1, then we know that, when we know that this R inverse is going to take the, wait I can actually put it, put it in here.

And if I do see R inverse C transpose, it will, like I said, it will not exactly diagonalize because rho is tending to 1, it is not equal to 1 because rho equal to 1 would be a be a degenerate case. So, when rho tends to 1, that is why the range of rho is between 0 and 1 not, not rho equal to 1 or rho equal to 0.

So, when you are, when you are kind of inching towards 1, when you come closer and closer that means when the correlation becomes higher and higher and higher, then you can show that in place of B, if I kind of put in this R inverse and R inverse has the structure of, structure of this, suppose we call this as B right? I mean, that is the kind of matrix that that a DCT can diagonalize.

So, you see that, so you see that what will happened is when you when you multiply R inverse on the left by C and on the right by C transpose, you'll get an almost diagonal matrix, it may not be exactly diagonal because of the fact that we still have, we do not have 1 minus alpha there. But as rho comes closer and closer to 1, you will start to see that this diagonalization capability begins to, begins to like, begins to look, look really good.

And we know that something, something is so which that means that the columns of C transpose then become the eigenvectors of our R inverse, correct. Kind of I mean, they are not exactly eigenvectors, but as I said right as rho tends to 1, these become more and more close to being the eigenvectors of, of R inverse. And you know that if something is (())(10:19) A has an eigen vector ei, and you know that Ai, Aei is some let us say lambda ei.

Now, if I phase invertible and suppose I do A inverse Aei then I have lambda at A inverse ei. Which then means that A inverse ei, it will be equal to 1 by lambda ei, right? So, it actually means that when something is an eigenvector for A inverse, it is also an eigenvector for, for A, correct.

So, in this case what this means is that, so, so this very same columns of C transpose will also, will also, you can claim that they will also be eigen vectors for R because our real interest is in R, we want to be able to write diagonalize R. So, it turns out that when you just look at R, it does not does not immediately apparent as to why these have to be eigen vectors for R, but then if you look at R inverse structure, R inverse structure looks very similar, very similar that looks like a symmetric diagonal matrix.

For this R, for this kind of R. And then, and then the fact is when rho comes closer and closer to 1 then it, then it kind of begins to look like the symmetric triangle that tridiagonal matrix that, that DCT can diagonalize and therefore, that is why we say that it tends to, tends to look like the KL transform, though, though, see in DC, a DFT case, when you had a circulant matrix, we said it is equal to the KLT.

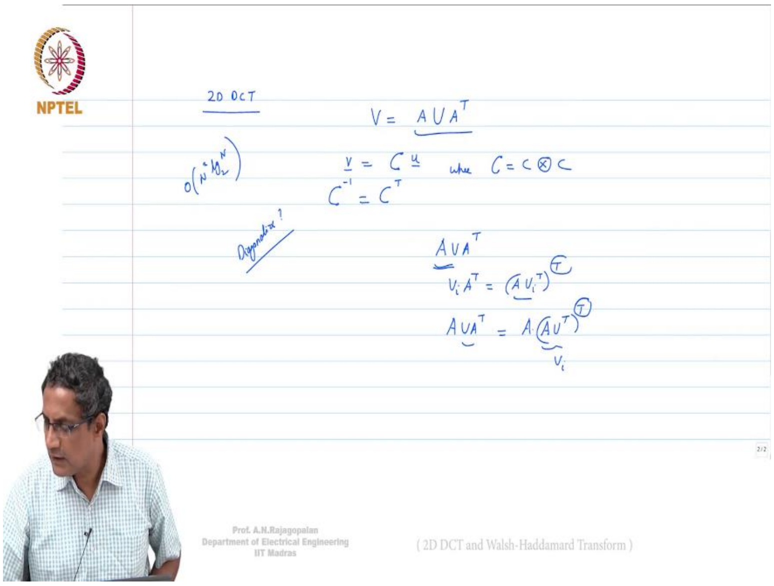
Here, we will not make the statement that DCT is equal to the KLT for this, for the first order Markov when rho tends to 1 but when rho tends to 1, it will, it will kind of, kind of come closer and closer to being the KLT. We would not say it is the KLT but then it comes closer and closer to but whereas a DFT cannot, cannot do this, (())(11:56).

So, suppose instead of C, suppose you put 5 there, suppose you put R 5R 5 transpose, these R, this kind of Markov thing and suppose I take R equal to, R now close to 1, let us say 0.95 or something, and so 4.9 even if you put that and then if you try to see right then, then what you will find is that the, the diagonalization capability of DFT will not be as good as actually a DCT.

But nice thing about this is, this is more a statistical property. So, it means that all signals that, that kind of satisfy, that satisfy, that kind of Markovian law, they are all likely to be I mean, so for all of them these eigenvectors are good, eigenvectors coming out of a DCT even though they are actually data independent.

Correct, I mean, right, DCT, we didn't choose depending upon R, we just chose a DCT basis. Turns out that that basis even though it is data independent, but then it is actually close to being optimal for this kind of matrices, for this kind of covariance. Any, any doubts at this point? Anyway, I am going to leave it to you to, to find out, I mean, and suppose I asked you what might be then the, then the really drew, Eigen sort of a basis for this R, I will leave it to you to find out. I have not done, I have not said that, it is said that this looks closer and closer. So, C has this, we know that C has this alpha k then a cos pi 2N plus 1k by 2N.

(Refer Slide Time: 13:31)



But if you really wonder what might be the, now coming to so the extension to a 2D sort of, sort of a DCT is, is straightforward. 2D DCT, so, so, it all extends the same way. We will not spend too much time, so you can think of a, I mean, think of a, think of an image that is given to you, which is you and then you want your V, so, we can of course write this as a AU, A transpose, we know.

And instead of A substitute C which is a DCT or in other words, you can get us an order it like we said the other day and then we can do something like a C script acting on you where what is, what is a C script now? C Kronecker C. And, and A, again and I leave it to you, you can easily show right that C inverse a C transpose, all that will follow on, C inverse is C

Kronecker C inverse, which is C inverse Kronecker inverse with the C transpose Kronecker C transpose b c canonical C transpose which is script C transpose.

So, all this, all the, we will not repeat all this, okay? We have done all, done this, now you are, so I am just writing in some kind of a different C, so that you can actually differentiate it from the usual C that we write for our 1D DCT. And in terms of the computational complexity, because you can, because yesterday we showed that you can compute it using a Fast Fourier Transform.

And therefore, the computational complexity for a 2D DCT is order N square log n to the base 2. The same as whatever you had for actually DFT. And then, and then this interpretation goes through for all transforms, like I said the other day, so when you have AU A transpose, you can either look upon this as A acting on the columns of U. See, and always going to remember that V and U, we have a certain interpretation, when U is kept like that M, whatever MN then we interpret K to be the, K to capture the variation along M and N to, L to capture the variation along N.

So, here you can think of A which is your whatever transform that you have acting on the columns of view to give you some this intermediate array UI, A transpose and then, so if you have taken first column, you should then, then take the take the rho, transform of the rho. So therefore, this is the same as AUi A transpose the whole transpose.

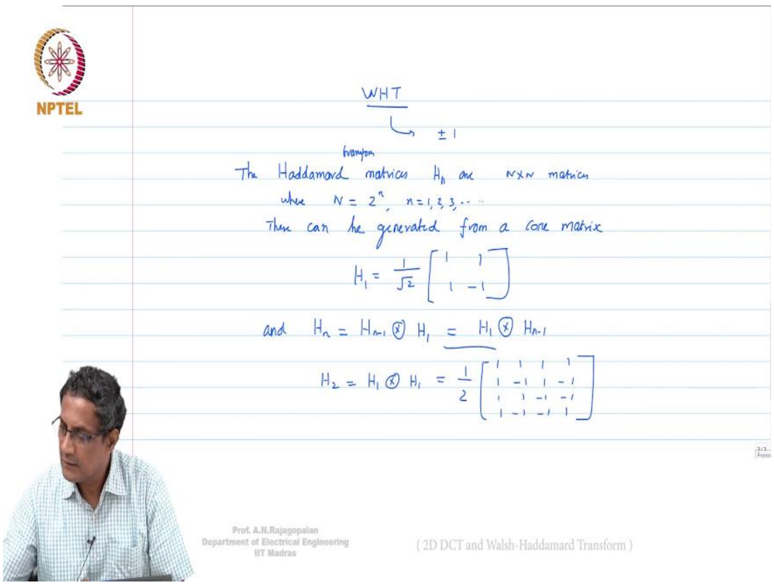
So, A UI transpose will take care of the fact that you will come to, you will take the then the rhos but then you should take again a transpose otherwise your V and U, the K variations along M and then L capturing variations along N, for that to happen, one more, you should get a re-transpose. Or you think about it as AUA transpose, you first do this guy, which means that this you will write as A and to what we write it as AU transpose, correct? AU transpose, transpose.

And then which now means that you are first taking the rhos of U. Correct, I mean, you are taking the transfer, you are already transforming U. So, that means you are coming, so UI, already taken the rows, now you should take the columns. So, the transpose will take the column and the transpose will also make sure that your K variation will capture, K will capture variations along M, N will capture variations along, L will capture variations along N.

So, so, that is why the next transpose will also get you to the rows, get you to the, get you to the columns. U transpose means, you are doing the rows first and then you're doing the columns and this transpose will also take care of the fact that V, and V and U have the same sort of interpretation.

The K, L interpretation for M comma N. So, in general this is true, so we know not every time we keep writing summation and keep showing, you can directly insert at any orthogonal transform which is separable and can be written as AUA transpose, all that separability will help you rho column, column rho whatever you want to do. Just said in each case, you may not specifically substitute the particular transform. And I am going to leave it to you as an exercise to show us to what kind of matrix will 2d DCT diagonalize. And all the other properties still hold, even for images, the fact that 2N by 2N, all that cyclically repeating on the grid, all that will still value.

(Refer Slide Time: 17:45)



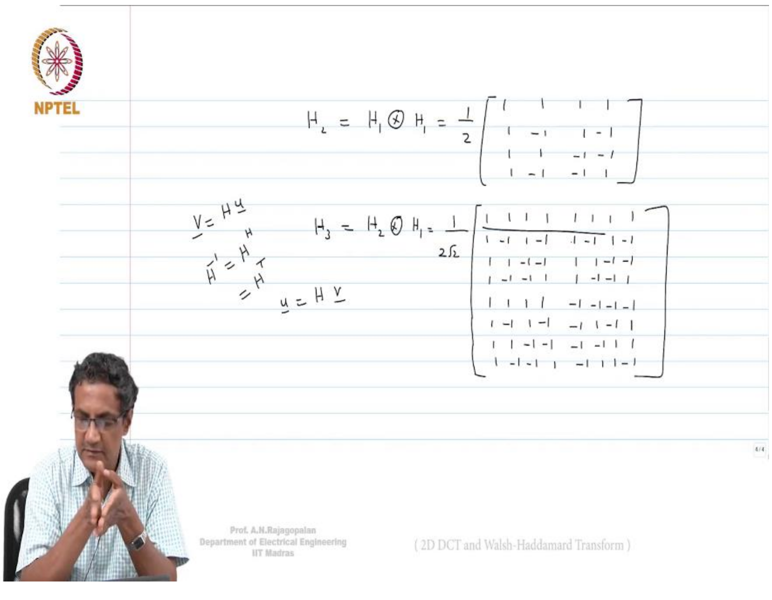
Now let us, move on to one more transform. So, we have seen a trigonometric transform, there are two of them, in fact, one is this DFT and another is DCT, so we will go to something which is not actually trigonometric. So, which is, what is the Walsh-Hadamard transform. The idea is not to do everything. In fact, if you read now, any of these transforms, you will know, you will know how they determine, how they work and so on.

So, Walsh-Hadamard transform. So, this, this actually the, in the, I mean, nice thing about it is the entries are kind of, see, plus minus 1, because there is a scale factor that sits outside, otherwise, all entries are like plus minus 1. So, everything that you want to do, you can just do in terms of, in terms of additions and subtractions, and so on. And so, the Walsh-Hadamard, the way you build it up, is to actually write it in the following form. So, so you start with actually a whole matrix or the Hadamard matrices, as they are called. Hadamard matrices, Hadamard transform matrices, of course, also matrices Hn, as they are called are N by N matrices, where N is a power of 2, n equal to 1 2 3 and so on.

And these are generated by, from a core matrix, these can, these can be generated from a core matrix H1, let us say, let us call this as H1, which is 1 by root 2 1 1 1 minus 1. So, at, the 2 cross 2, this looks like a DFT, this looks like a DCT, they are all identical but as you get higher, I mean higher sizes then they would not look.

And any, any Hn that you want is simply Hn minus 1 Kronecker H1, which also in this case, you can even write it as H1 Kronecker Hn minus 1, because, because everything is coming H1 Kronecker with itself. For example, H2 is what H1 Kronecker with H1, and therefore this we can write, H1 Kronecker, H1 will then become 1 by 2, 1 1 1 minus 1 1 1 1 minus 1 1 1 1 minus 1 minus 1 minus 1 minus 1 1. And then if you want to go to lets say H3, you would not go beyond that.

(Refer Slide Time: 20:30)



So, H3 is H2 Kronecker H1. So, which is nothing but 1 by 2 root 2 now, because, and then what is it? So, 1 1 1 1 1 minus 1 1 minus 1 1 1 minus 1 minus 1 1 minus 1 minus 1 and H3 is H2 Kronecker H1, which is 1 by 2 root 2 1 1 1 1 1 minus 1 1 minus 1 1 1 minus 1 minus 1 1 minus 1 minus 1 1 1 1 1 1 1 minus 1 1 minus 1 1 1 minus 1 minus 1 1 1 1 1 1 1 minus 1 1 minus 1 minus 1 1 1 minus 1 minus 1 minus 1 1 minus 1 minus 1 minus 1 1 minus 1minus 1 minus 1 minus 1 minus 1 1 minus 1 1 minus 1 1 minus 1 1 minus 1 minus 1 1 1 minus 1 1 1 minus 1. With look, so, so if you see here, you see an H3 and so on.

So, we can actually, we can go up and incidentally it turns out that if you try to do V is equal to Hu, suppose, suppose somebody gives you a sequence u of whatever length, and suppose you do Hu then, then H inverse, which you would expect to be H Hermitian because these are all orthogonal transforms H is Hermitian and then in this case its real and then you can also, you can also check that H transpose is equal to H. Therefore, in fact u is equal to simply H times V. So, H inverse is simply equal to H. So, so in that sense, because, because its kind of binary valued except for that factor that is outside.

So, so it can be implemented fast. So, there are fast versions again, but then thing is it does not have the kind of energy packing capability and all like that of a kind of DCT. Therefore, there is not really a preferred kind of transform, but then for hardware implementation or that its good because you just, you are just dealing with binary values. So, that way, the hardware implementation part for this, for this and its also called Walsh because if you look at the rows of this matrix, these are, these are Walsh functions.

So, there are these Hadamard matrices and then when you, when you do, do a Kronecker recursion then you actually get, get this, get these, get these rows or columns, they are both the same. So, if you look at them, they are kind of Walsh functions. That is why its called a Hadamard Walsh transform.

And again, all the, all the things are still good in the sense that fast computation, all that is fine except that there are no, there are no grade or statistical properties and all for this. The implementation part is good to implement it in hardware. Now, with this, anyway, going to 2D is all very simple, just do an H Kronecker H, whatever it is on the, on this H, and then you will, it will take you to 2D.