Distributed Optimization and Machine Learning

Prof. Mayank Baranwal

Computer Science & Engineering, Electrical Engineering, Mathematics

Indian Institute of Technology Bombay

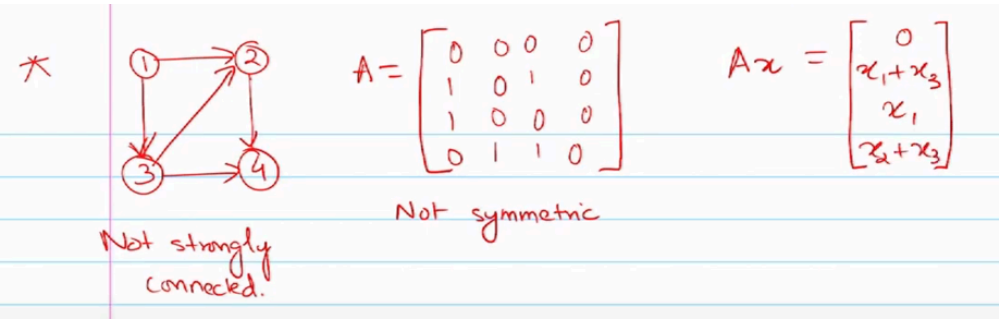
Week-8

Lecture 28: Consensus and Average Consensus

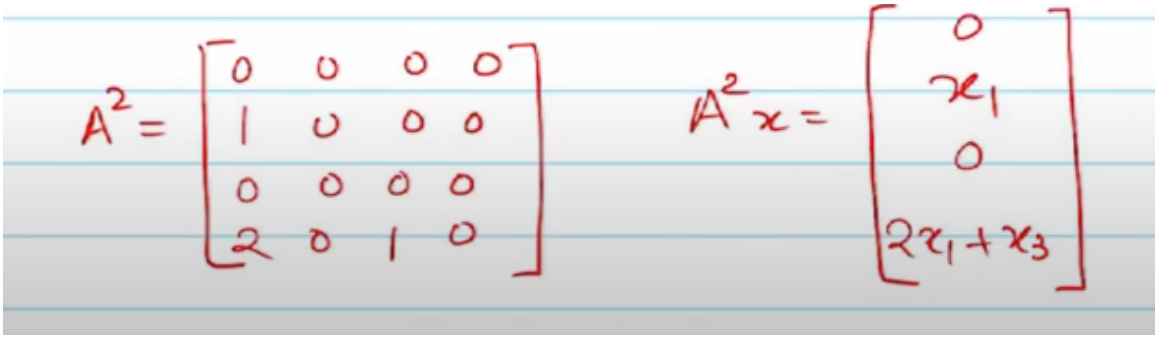
So, everyone now knows what adjacency matrix in a graph is right. So, let us try and see what the role of adjacency matrix is. So, let us say I have a graph which looks something like this. So, it is a directed graph. you can for now assume that the edge weights are simply 1. So, first of all is this graph strongly connected? No right, so not strongly connected.

Ok so how, so what does the adjacency matrix for this graph looks like? Are there any self loops? So these diagonal entries are going to be all zeros. So they are only outgoing edges let's say from 1, they are only outgoing edges to 2 and 3 right. So that means x can, 1 can broadcast to 2 and 3, but cannot receive any message from anyone else. So it can broadcast to 2 and 3.

So this is the entry for the Likewise you have for 2 it can broadcast to 4 and you have 0s here, for 3 it can broadcast to 2 and 4 and 4 it cannot broadcast to anyone, so all the entries are going to be And as I said in case of directed graphs, it is possible that the adjacency matrix is not symmetric and that is what you see right. So, not symmetric. So, when I do a times let us say x, x being x 1, x 2, x 3, x 4. So, what is the output of this? So, that would be 0 for the first entry. because if 1 is not going to be getting any information from anyone right, what about 2? So, x1 plus x3, this is going to be x1 and this is x2 plus x3.



So, let us look at what happens when we use A square. So, what is A square? You can do the math, it turns out that A square is simply So, this is your A square. Now, if I do A square times x, what does this look like? So, the first entry is again going to be 0, the second entry is x1, 0 and 2x1 plus x3. So, I mean this makes sense because you can see that 2 is going to be receiving information from let us say here. So, 2 is going to be receiving information from 1 and 3 and that is because of this right, this edge and then this particular edge right.

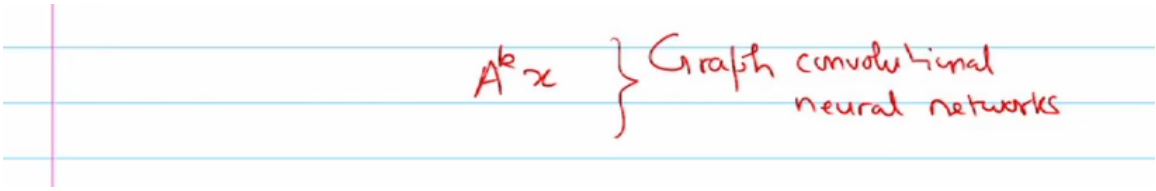


So, you have x 1 and x 3 showing up. Likewise, 4 is going to be receiving information from 2 and 3 and that is why you have x 2 and x 3 showing up. Why do you have x 1 showing up here? for that matter x 1 and x 3 showing up here. So, if I look at the second one right 2. So, 2 yeah.

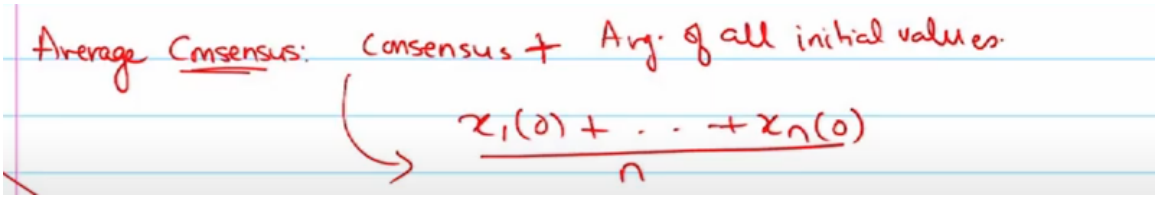
So, in this case we have x 1 showing up and actually there is a 2 hop information flow from 1 to 2 right. And if I look at the x 4 there is also a 2 hop information flow from here and 2 hop information flow from here ok. So, that means if I do A to the k times x. So, what I am essentially doing is I am aggregating information for my k hop neighbors or I am receiving information for my k hop neighbors. right and that is in fact how the information propagates in a graph.

So, the first time I do A times x, I basically receive information from my immediate neighbors But in that process, the neighbors would have also received information from their neighbors, right? So the next time when I aggregate information from my neighbors, I'm going to be receiving the information that my neighbors have aggregated from others, right? So essentially getting information from my two hop neighbors and three hop neighbors and so on. So that's how the information flows in a graph and that's why when you have a line graph, where you then for the information to propagate from let us say the first node to the end node that is going to take at least 10 minus 1 steps because that is that is the amount of time it is going to take to propagate information from one end to another right. But then so if you guys are aware of something called graph convolutional neural networks or if you are aware of neural networks, I mean like are you guys aware

of neural networks in general or convolutional neural networks? So, convolutional neural network is that you aggregate information from your neighbors. So think of this particular application as information aggregation with your one-hop neighbors, two-hop neighbors, so every layer in fact A to the k when you say A to the kx that means there are k such layers, every layer is aggregating information from one-hop neighbors, two-hop neighbors, three hop neighbors and so on and in between you introduce some level of non-linearity and that's how you make any predictions on graphs if you have graph structured data. So that is the idea.



So, this brings us brings us to an important topic in the context of distributed optimization which is consensus and by definition basically it amounts to all nodes converging to a common value. So, consensus just talks about every node eventually converging to the same common value that common value can be anything. I mean you can like let us say all nodes can eventually converge to 0 being, but that may not be very useful information. So, sometimes not it is not just the consensus part that we are interested in, but we are also interested in something called average consensus. So the idea is every node, so this is consensus, so every node converges to a common value so that is there, but that common value happens to be average of all initial values.

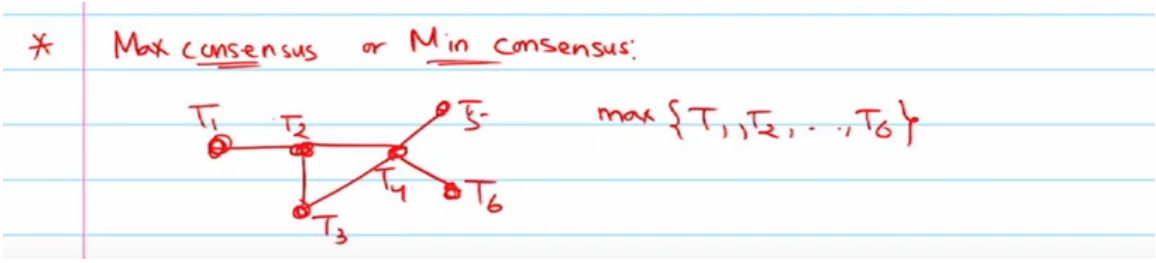


So essentially if you have x1, x2, x3 and x4 as the value that every node possesses, and if there are n such nodes. So, you want consensus algorithm to essentially arrive at this particular common value. So, this is called average consensus ok. So, you have a belief someone else may have a belief and you want to arrive at the average of all the beliefs and this is what average consensus is about. So, if you remember the example that we considered in the first lecture itself which was about temperature sensors being installed and every every every sensor measuring temperature differently and we want to arrive at the average of all what everyone else is measuring right.

every sensor is measuring and we looked at two different ways to run the consensus algorithm and it so turned out that in one case we were able to arrive at consensus but it

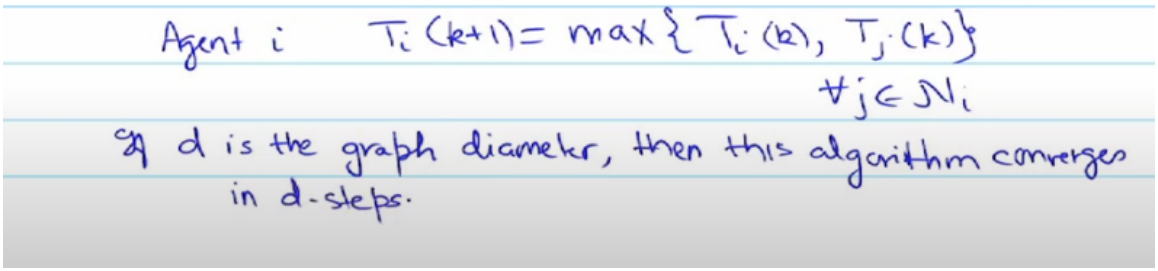
was not the average consensus that we intended to and by choosing a different consensus algorithm or different weighting scheme we were able to arrive at the average consensus. So the goal for today's lecture is to understand the conditions under which you, so conditions under which consensus is achieved and then conditions under which average consensus is achieved. So, you have to have certain conditions if those are satisfied then you can guarantee average consensus, and so on. So, we are going to study those sufficient conditions.

conditions. There are other forms of consensus that you may often find in other papers which are max consensus or min consensus. And as the name suggests, Every agent tries to arrive at the max of the values that like every agent in the network processes, right. So, let us say you have a graph which looks something like this. And agent 1 has a value t1, t2, t3, t4, t5 and t6. So, the objective of max consensus is that every agent is able to arrive at the max of all these values t1, t2, t3 So, can you think of an algorithm which does that? So, let us say I want to achieve this max consensus in this particular graph.



So, what kind of algorithm would a very simple algorithm that you can think of that can achieve max consensus or min consensus for that matter. Again agents can only communicate with their neighbors right. So, that is that is one of the constraints that the agents have. It is very simple yeah. So, every agent just exchanges information and basically replaces its current value with the max of whatever information it has received from its neighbors right.

And that would and just replaces its current state with the max of its current and its neighbor state right. So, agent i. So, Ti k plus 1 is simply going to be max of Ti k and Tj k for every j in the neighbor would certify right. So, every agent runs this and how in how many steps is it guaranteed to converge? Yeah, so the diameter of the graph right. So, if diameter of the graph is d is the graph diameter then this algorithm converges in in d steps ok.



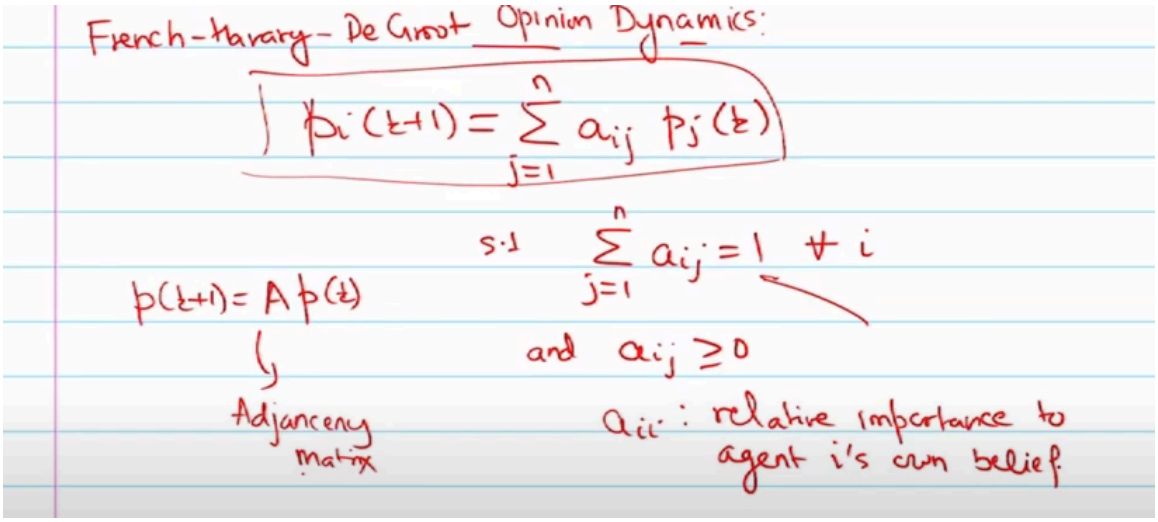
That may not be the case with the with the other can like asymptotic or the average consensus things that we algorithms that we are going to talk about. There is an naive way to let us say obtain average consensus is that you have you run the max consensus I mean you also. So, essentially you are going to be storing the max value, but then at the same time you also have a copy of your current value right. So, in the first d step you are going to be storing the max value.

in the second d steps you are going to be storing the second largest value and the third largest value and the fourth largest value and so on. So that means in n times d steps right you would have essentially all the values known and then you just take the average of it and that is also going to give you the average consensus. So that is a naive way to go about it and in fact in a finite number of steps you are guaranteed to get the exact average consensus value, but that is not the kind of algorithm that we are going to be looking at because if the graph diameter is large you do not want and if or let us say the graph diameter is also small, but if number of agents is very large you cannot just wait for n times t steps right. So that is so we would want to avoid this kind of naive way to. obtain average consensus or the consensus.

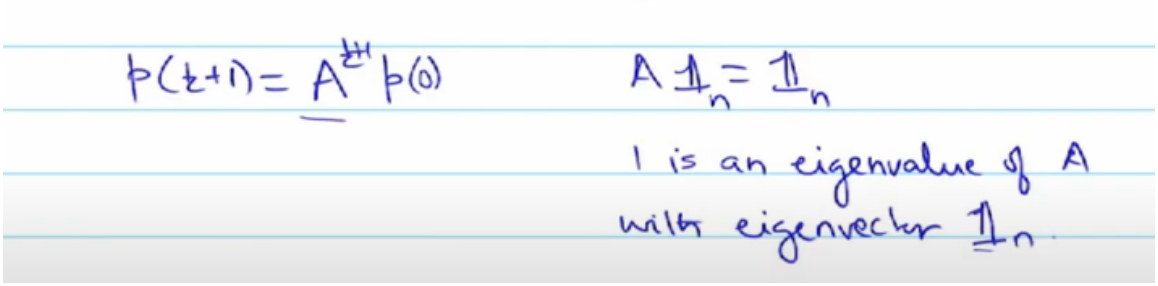
So, this kind of scheme is often used consensus often used in opinion dynamics as I mentioned that you have certain beliefs, your neighbors may have certain like some other beliefs and you want to maybe obtain like the consensus of all the beliefs right. Let's say I mean you vote for one particular party, your friend votes for some other party and so on right and you want to see where the average trend is going on. So, that this is where the average consensus or the consensus models really help you with. So, in this context you have the French Harary De Groot dynamics model, opinion dynamics ok. And the way it works is that your opinion at time t plus 1 or a step k plus 1, the dynamics essentially say there are n agents in the network a ij where such that ok.

And let and a ii is nothing, but the relative like relative importance to its own belief So, what let us let us look at this dynamics in sort of more detail. So, aij potentially can be 0 that means j is not a neighbor of i right, but if they are non-zero then you look at the your neighbors belief and you essentially do a weighted average of those neighbors belief and this is your current belief now ok and then you sort of run it multiple times. So, if you

look at the dynamics of this, this is nothing, but p of t plus 1 is the is essentially A times p of t, where A is the adjacency matrix of the graph and this time it is going to be a weighted graph, okay. So, what property does A have here, this matrix A? So, A is row stochastic, right. So, that means the row sum is 1.



So, A is row stochastic, row sum is 1. So, does row stochastic A guarantee consensus? So, that is the first question that we are going to look at and the second question is does row stocastic A guarantee average consensus ok. So, what do you think the answer should be for both these questions. So, p of t plus 1 is essentially A to the p of t plus 1 let us say p 0 right ok. So, what do we know about A? A is row stochastic meaning A times if I look at vector of all 1's this is going to give you vector of all 1's right.



Let us call it let us say they are n agents. So, this is what it is going to give you. So, that means this vector of all 1's is is an eigenvector of A with eigenvalue also equal to 1 . So, as long as as long as the eigenvalues of other eigenvalues of A are strictly smaller than 1 or let us say smaller than 1 ok or less than equal to 1 that means your opinions are I mean not diverging right as you run this multiple times as you make a to the t plus 1 as you are exponentiating it further and further So, I mean think of it in the scalar case right, let us say if it is just a scalar case if it I mean eigenvalue less than 1 or less than equal to 1 means that you are not sort of diverging the beliefs with every iteration right. So, then

you can hope that in fact this would arrive at consensus.

So, with row stochastic A you can guarantee consensus. So, this is true, but with row stochastic A you can you need not I mean you would not be able to guarantee average consensus. So, this is not true. in general I mean it is possible, but it is not true in general ok. So, let us let us revisit the example that we looked at in the first lecture. So, it is a simple setting that we are looking at. So, you have a graph in this case you have 4 different values. So, there are 4 different sensors each measuring temperature. So, one of them is one of the sensors measures 25 Celsius, 20, 24 and 27. and the goal is to arrive at average consensus. So, average of these senses right and the connectivity that we have is 1 is connected to 2 and vice versa, 2 is connected to all the senses, 3 is connected to 2, 3, 2 and 4 and 4 is again connected to 2 and 3. So, that is the connectivity that we have right. So, if I consider this particular adjacency matrix A 1 and I run this algorithm.

```
x0 = np.array([25., 20., 24., 27.])
A1 = np.array([[0.5, 0.5, 0., 0.], [0.25, 0.25, 0.25, 0.25], [0., 1/3, 1/3, 1/3], [0., 1/3, 1/3, 1/3]])
```

So, basically essentially doing A times x and then we are just kind of repeating this over right. So, we see that it eventually arrives at a consensus value, but then it is not the average consensus, the average value is 24 whereas this arrives at something at 23.

```
[array([25., 20., 24., 27.]),
 array([22.5       , 24.        , 23.66666667, 23.66666667]),
 array([23.25      , 23.45833333, 23.77777778, 23.77777778]),
 array([23.35416667, 23.56597222, 23.6712963 , 23.6712963 ]),
 array([23.46006944, 23.56568287, 23.63618827, 23.63618827]),
 array([23.51287616, 23.57453221, 23.61268647, 23.61268647]),
 array([23.54370419, 23.57819533, 23.59996839, 23.59996839]),
 array([23.56094976, 23.58045907, 23.5927107 , 23.5927107 ]),
 array([23.57070441, 23.58170756, 23.58862682, 23.58862682]),
 array([23.57620599, 23.5824164 , 23.5863204 , 23.5863204 ]),
 array([23.5793112 , 23.5828158 , 23.58501907, 23.58501907]),
 array([23.5810635 , 23.58304128, 23.58428465, 23.58428465]),
 array([23.58205239, 23.58316852, 23.58387019, 23.58387019]),
 array([23.58261045, 23.58324032, 23.5836363 , 23.5836363 ]),
 array([23.58292539, 23.58328084, 23.58350431, 23.58350431]),
 array([23.58310312, 23.58330371, 23.58342982, 23.58342982]),
 array([23.58320341, 23.58331662, 23.58338778, 23.58338778]),
 array([23.58326002, 23.5833239 , 23.58336406, 23.58336406]),
 array([23.58329196, 23.58332801, 23.58335067, 23.58335067]),
 array([23.58330998, 23.58333033, 23.58334312, 23.58334312]),
 array([23.58332016, 23.58333164, 23.58333886, 23.58333886])]
```

5 it is something right. And if you look at the matrix A, so is this row stochastic? Yes right. So, it is row stochastic. So, consensus I mean you see the consensus happening, but there is no average consensus. But if I change my A to be to look something I mean which looks something like this now right.

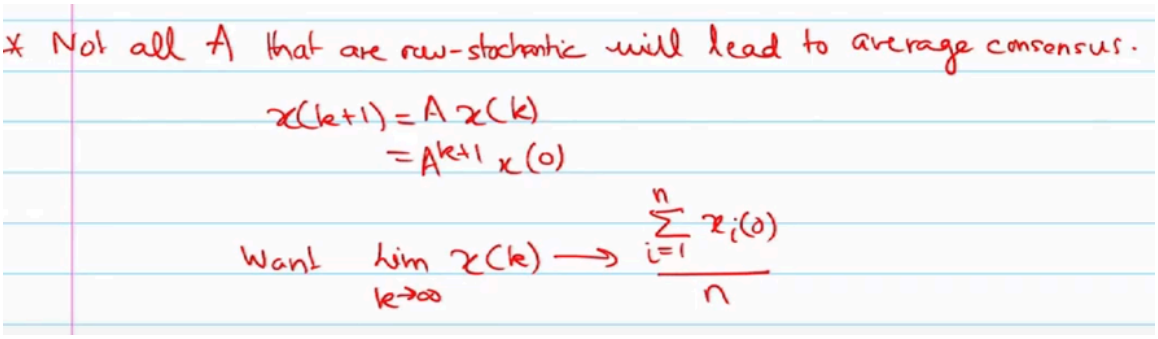
```
x0 = np.array([25., 20., 24., 27.])
A2 = np.array([[3/4, 1/4, 0.,| 0.], [0.25, 0.25, 0.25, 0.25], [0., 1/4, 5/12, 1/3], [0., 1/4, 1/3, 5/12]
```

So, again it is row stochastic and it is not just row stochastic it is something else as well. It is symmetric as well right and this also makes it column stochastic. So, if I run this with this particular choice of A you see that eventually they arrive at the consensus value right ok. So, which is the average consensus value which is 24.

```
[array([25., 20., 24., 27.]),
 array([23.75, 24.  , 24.  , 24.25]),
 array([23.8125    , 24.        , 24.08333333, 24.10416667]),
 array([23.859375  , 24.        , 24.06944444, 24.07118056]),
 array([23.89453125, 24.        , 24.05266204, 24.05280671]),
 array([23.92089844, 24.        , 24.03954475, 24.03955681]),
 array([23.94067383, 24.        , 24.02966258, 24.02966359]),
 array([23.95550537, 24.        , 24.02224727, 24.02224736]),
 array([23.96662903, 24.        , 24.01668548, 24.01668549]),
 array([23.97497177, 24.        , 24.01251411, 24.01251411]),
 array([23.98122883, 24.        , 24.00938559, 24.00938559]),
 array([23.98592162, 24.        , 24.00703919, 24.00703919]),
 array([23.98944122, 24.        , 24.00527939, 24.00527939]),
 array([23.99208091, 24.        , 24.00395954, 24.00395954]),
 array([23.99406068, 24.        , 24.00296966, 24.00296966]),
 array([23.99554551, 24.        , 24.00222724, 24.00222724]),
 array([23.99665913, 24.        , 24.00167043, 24.00167043]),
 array([23.99749435, 24.        , 24.00125282, 24.00125282]),
 array([23.99812076, 24.        , 24.00093962, 24.00093962]),
 array([23.99859057, 24.        , 24.00070471, 24.00070471]),
 array([23.99894293, 24.        , 24.00052854, 24.00052854])]
```

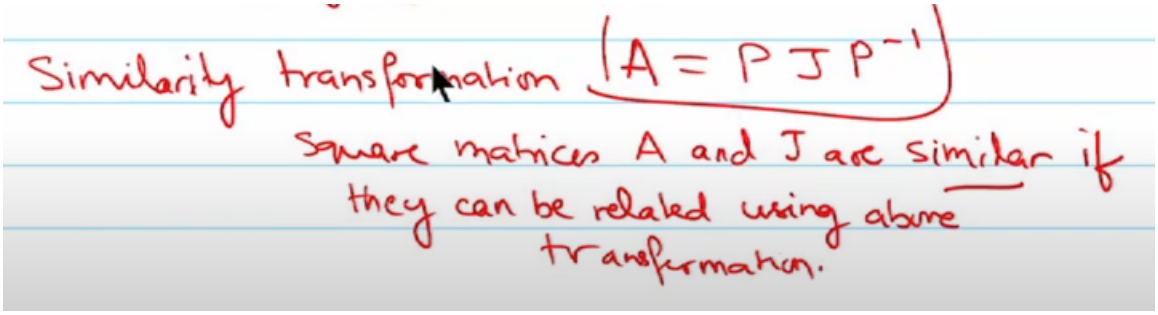
And so, in this lecture we are going to look at the theory behind it as to why doubly stochastic matrix or the basically guarantees you average consensus, ok. So, the point that we are trying to make here is, so not all A that are row stochastic, will lead to average consensus.

So, basically what do we want? We want that x k plus 1 which is essentially A to the A times x k. So, we have or another way to write this is A to the k plus 1 times x naught. So, we want limit k goes to infinity. x k is essentially goes to summation this is what we want. So, we want that all agents they converge to this common value.

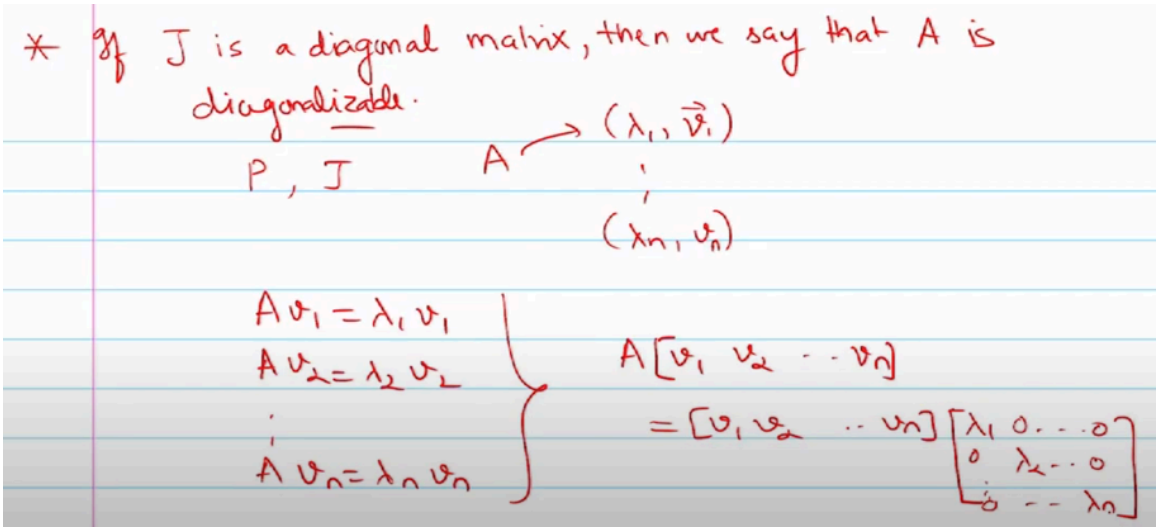


So, for this we would need a little very brief refresher on linear algebra. So, let us briefly revisit linear algebra to be able to understand this in more detail. So, everyone here had has had some course on linear algebra. So, similarity transformation, diagonalizability is that ok with everyone.

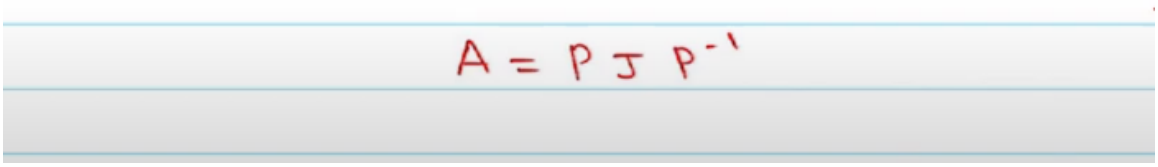
So, we will just quickly recap this. So, what is similarity transformation of? So, if a matrix A, so what do you mean by similarity transformation? So, which is? So, something like this right PJP inverse right. So, we say so, square matrices first of all A and J they need to be square matrices right. So, in fact p and p and p as well. So, A and J are similar if they can be related using this particular transformation ok.



So, if J happens to be a diagonal matrix, then we say that A is diagonalizable.
We will come to that P part. So, what is what could be a good choice for P and J? So, how do we choose P and J here? So, let us say you have let us say A has n different eigenvalues eigenvector pairs. So, A is n cross n and it has these eigenvectors eigenvalue pairs right. So, we know that A v 1 for instance is lambda 1 v 1, A v 2 is lambda 2 v 2 and so on. And for now let us just assume that lambda 1 through lambda n are different ok. So, I can write this as A times v 1 v n ok.

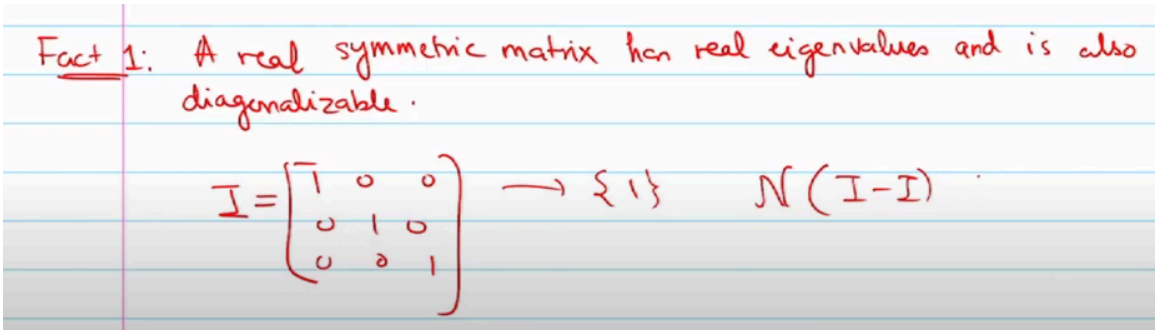


And this you can call the J, this is your P and you can write this as P J P inverse.

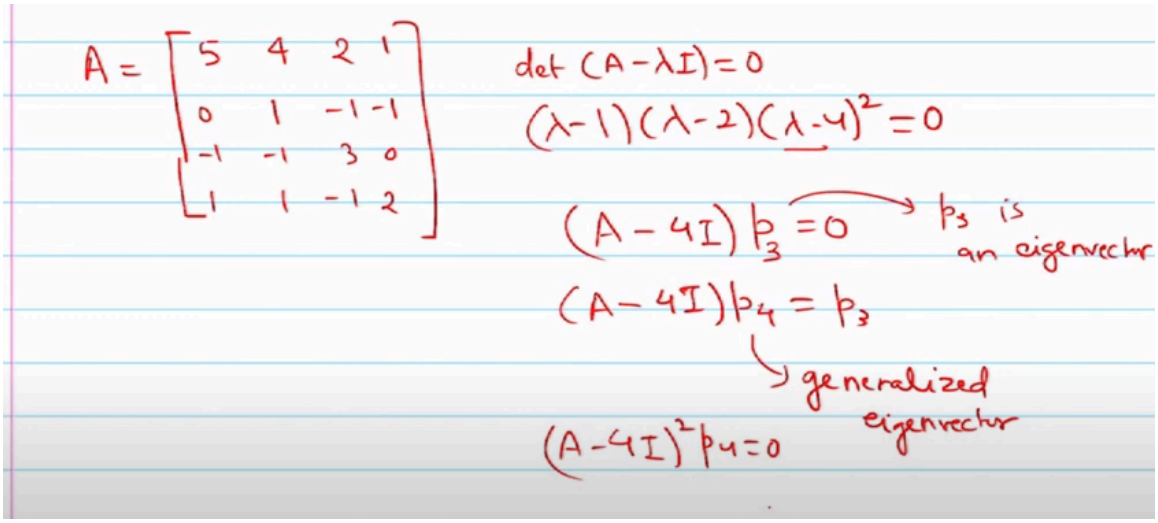


If you have n different eigenvalues and n different eigenvector pairs, they are going to be linearly independent, P inverse exists and therefore you can write A as P J P inverse ok. So, we assume obviously A has distinct also let us say simple eigenvalues. So fact one, so we are going to be here. So the fact one is, so a real, by the way when we talk about doubly stochastic it is going to be a symmetric matrix.

So one of the facts is that a real symmetric matrix has real eigenvalues and is also diagonalizable ok. So, when we talk about let us say in something like identity matrix which is let us say 3 by 3 So, what are the eigenvalues of this matrix? So, all all eigen like so eigenvalues are essentially I mean there is just one eigenvalue right, but what are the eigenvectors? So, what is what is the definition of an eigenvector? So, how do you find the eigenvector if you know the eigenvalue? Null space of A minus lambda I right. So, null space of A minus A in this case a is I simply identity. So, identity minus identity right which is a 0 matrix right. So, essentially in fact these three columns are the eigenvectors.



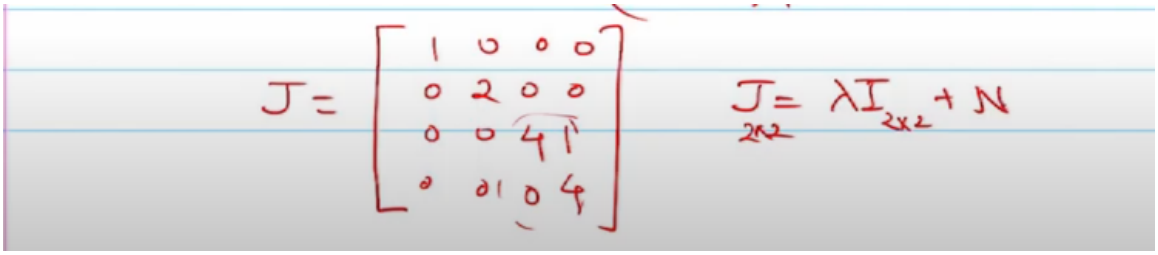
So, even though you have one eigenvalue you have distinct eigenvectors right. So, whereas if you consider an example which looks something like this. Let us say 5 this particular matrix 5 4 2 1, 0 1 minus 1. So, how do we find the eigenvalues of a matrix? Just set the determinant of A minus lambda, just find the characteristic equation. set it to 0 and it would it turns out that the Eigen basically the characteristic equation is given by lambda 1 minus lambda minus 2 lambda minus 4 whole square equal to 0.



Now, the repeated Eigen value is lambda equal to 4 right and it turns out that if I try to find the null space of a minus 4 identity like if I try to find a vector p such that this is equal to 0. unlike in the previous case where I actually get three different vectors even with in this case I am going to get just one vector. I mean plus and minus the scale scaling, but I am going to get just one vector right. So, then what do we do? Square what? So, what is it called? So, let us say I get p.

So, p 3 is my p 3 is an eigenvector. corresponding to 4, eigenvalue 4. So then what do we do? So we find P4 such that this is equal to P3. So this is called generalized eigenvector. which is same as I mean if I multiply this by A minus 4 I times identity then this is equal to 0.

So, essentially you square it and try to find it. So, when we try to diagonalize this kind of, so first in this case this a is not diagonalizable, it is not perfectly diagonalizable, but it is I mean it can be approximately diagonalized using something called Jordan blocks right, is everyone familiar with Jordan blocks? No, ok. So, Jordan blocks are the like. So, essentially in this case J would look something like this. So, for all the eigenvectors for all the simple eigenvalues which have which do not have this multiplicity.



So, which in this case are 1 and 2. So, you would get something like this, but for the other one I mean technically if the matrix was diagonalizable this is what I mean you would have gotten zeros over here, but you get a 1 here and this is. So, you get a Jordan block. So, Jordan block J is essentially lambda i. So, in this case it is a 2 by 2 Jordan block lambda i 2 plus 2 plus a nil potent matrix.

So, this is this is a Jordan block here. So, if let us say let us say we were talking about 5 by 5 matrix dimensional matrix A and let us say 4 was repeated 3 times and still I mean and there was just one distinct eigenvector, then that means there are two generalized eigenvectors and you would have gotten this to be your Jordan block ok. So, again A plus lambda I plus n the Jordan the nilpotent matrix. But the point is if using these P 1 through P 4, so if I use P J P inverse with this being your J, you can still write A in terms of P like using similarity transformation as P J P inverse where J is not perfectly diagonal, but approximately diagonal. Thank you.