Numerical Methods and Programming

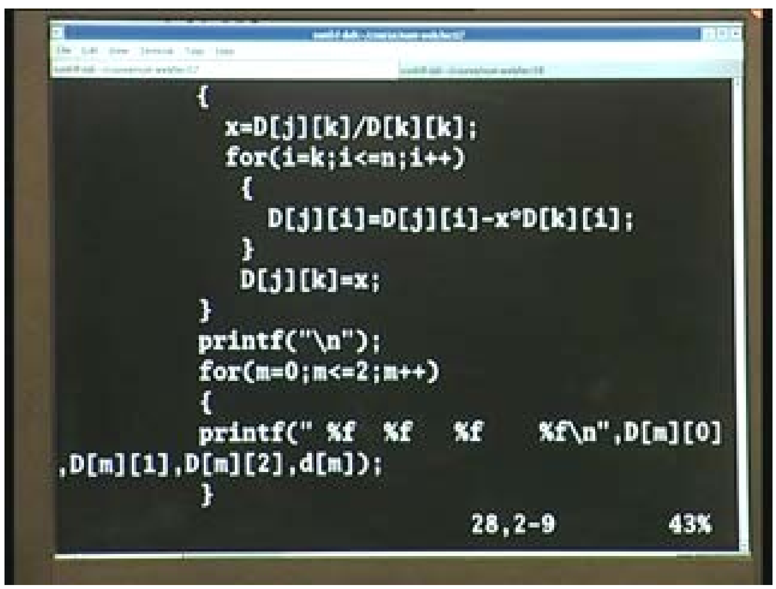
P. B. Sunil Kumar
Department of Physics
Indian Institute of Technology, Madras

Lecture - 18

Matrix elimination with pivoting and the condition number

In the last lecturer, we discuss the LU decomposition of matrixes. So we just summarize what we learnt there that is we, if have matrix equations are the linear, set linear equations of the form a x equal to b and b is solutions right side matrix which has some numbers and x is the unknown and a is matrix of the coefficient of the element, in the linear equations and then we have LX equal to b and then we can split the matrix a x equal to b and then, we can split the matrix "a" into 2 matrices l and u, one its lower triangular and one its upper triangular and then we can right this LU this matrix l as LU and that would transform the equations into LU into x equal to b right, so l is said to be a lower triangular and u is upper triangular and then we split the equations into 2 now, that is he could call you x equal to y examiner equations and then solve for LY equal to b.

(Refer Slide Time: 01:30)



So we can solve since we know the b and the l, we learn how to actually construct the l and u matrix given the a matrix and then, we can solve for this LY equal to b and then used the solution y, UX equal to y solve for x that is what we learn the last class. And we learn that, when you eliminate a to make upper triangular matrix by gauss eliminate process. We actually construct what we construct has the upper triangular matrix in the gauss elimination process, is actually this u matrix and then elements, we use to multiply the rows and then we subtract from the rows below that to make this column 0, this column elements 0 is actually the elements of the of the l matrix, that what be so and then the programme which actually implement that.

And then, we saw this program and which we actually implemented this idea that wherever okay, what our the using has for elimination process, okay we just store them as, so we store that factor which be use that to eliminate the column elements store the column elements itself. So we know replace a matrix replace is those what our this elements was the a matrix, now the u matrix 0. So I can use that the positions to store the elements of that l matrix so this positions to store in the elements of the l matrix because , we do not need the store the elements of the diagonal elements of the l matrix because we take them to be 1 choice which made an d that choice, we know that, this elements we simply factors to used the elements this elements and that is what so on here its that factor was x which is use the simply store in that the column available it.

And th en, once we have this matrices and we could first do LY equal to b because, we know the b which is right hand side of the matrix equations and, l is the upper is it lower triangular matrix. So we can do forward substitution $Alfa_{11}$ is 1, so the, y_1 simply b_1 and then $Alfa_{21}$ into $Alfa_{22}$ into $Alfa_{22}$ is 1 and $Alfa_{21}$ no 1. So we can do a forward substitution and get the elements of the f of y okay that its can implemented here as sequence I go from i equal to 0 to i equal to n, that is the number of y value which is its get y of 0 to y of n and then, we say that j the another variable which goes to from j to i minus 1. So when i equal to 0 that its only one element j is also 0.

So that its only one element, so than will say that x, now simply turn out of the x its initialize 0, so x is equal to d_{00} right i0, j0, d_{00} and then, so when i equal to 0 and j is also 0 doses in actually go into eliminate all side it does in go into particular line comes here and y of 0 will be simply d of 0 minus 0 x its initialize 0 right but, when 1 i is 1 now, this summarize the i to j equal to 0, j equal to 0, there is no one element here, i is 1 okay, then will be one substitution here okay, that is 1 execution of this command there it is, x goes from, x goes now from 0 to d_{10} okay, j is 0, i is 0, d_{10}.

So now, we remember d_{10} is actually this element $Alfa_{21}$ okay, which is store of $Alfa_{21}$ in the original matrix these are original matrix, say that we are store now this, the $Alfa_{21}$ has d_{10} right. So that say limit okay, that is d_1 its 0 into this one solve we cant ok so this process we will get and than subtract that from d of 5 and then the y of 1 again d of 1 minus d_{10} into y of 0, we already know y of 0 so this command basically means y of 1 equal to d and minus d_{10} into y of 1 right. So that is basically solving second line of the equation $Alfa_{21}$, $Alfa_{22}$ into y_1, y_2, y_1 y_2 is equal to d_2 that is what we are solving here okay, this are all my programme starts 0 actually become as d_{00}, 1 $Alfa_{11}$ equal to y_0 into y_0 y_1 equal to d_1 just we solve.

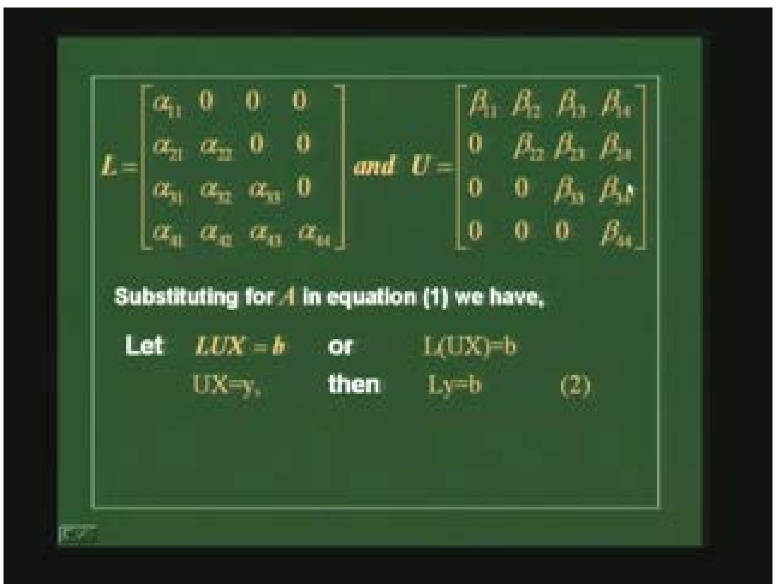
So we solve that, then the forward substitution, similarly and we can get all the y values from this and once you have the y value, we go to the next equation that is, we solve the y values and then we go to this equation and this we can seen earlier in the Gauss elimination process, this so that is upper triangular matrix and these are unknown and this we node now, y value so now we today back substitutions so we go back word so we actually back word since solve its know i starts the into 0 in the forward substitution the number go from 0 to n ok now we start from back substitution i goes from n to 0 and than we get the solution values and that is, what we have seen earlier.

So this way, we do LU decomposition the advantage of this with the elimination process we saw earlier is that in the elimination process we only keep this matrix that is u matrix right basically because, we take full a matrix a matrix and then eliminate this column and make the upper triangular matrix but than in order to keep the equation the same also need to change the written side of the matrix so the b value also to be properly changed.

So when we change a matrix keep the upper triangular ,we also make changed b value and that means every time referral equation, we have from a is the same but b is the different then we have to find new upper triangular matrix. So that is a kind of waste of time especially in cases where we have find the inverse of the matrixes see that later. So this case once we have, LU decomposition simply change in right hand side of the matrix, we can get difference solution for example, in the case we want to find inverse of the matrix, this is extremely useful as we will see as so.

Okay, so now, that is complete story, the LU has been written as, a has been written as L and U and we have use that to solve the equations. Okay so now, how we before we go into actually use of this matrix to find the inverse of the matrix, we will just look at the implementation, one implementation which is not a seen earlier that is in the gauss elimination, the everything, okay so one thing we have not seen is when you have matrix line this and then we want to eliminate all the column to see okay and make the upper triangular matrix using gauss eliminate scheme, we want to work, we multiply, we divide this row by this element, this diagonal element a_{11} and multiply by a_{21} and then etcetera from this, that is what to make it as 0, we saw this again.

(Refer Slide Time: 03:18)



So in this, we saw there is one drawback of this process, that problem is that, if any of the diagonal element tend to be small and then this factor which used multiply this row in this case for example, we want to the first column the factor which multiply this row of which is a_{21} by a_{11} will become a very large and as itself any small round of a for they amplifier so we saw that into method would be completely failed, if any of this diagonal elements goes to 0.So let us say for example, the process of eliminate this number which that is column elements which you want to do let its which goes is to 0 and next step do want to you element this column and below the second row we will find we done have the factor because the factor goes to infinite.

So in that case, we said that one way to solve this problems is to actually shift the rows, if any of this diagonal elements goes to 0, let say you given a matrix the diagonal element 0, and then you ask to make upper triangular matrix and then we come what will do normally is to multiply this row by a_{21} by a_{11} and a_{31} by a_{11} etcetera and subtract but you cannot do that a_{11} is 0. So what you do in that case is switch it, find out which is the largest element in this column right, if are which is largest element in this column and corresponding row is switch to the first one, if it end which change rows, yes its similarly the interchanged by right hand side two, so that is called the pivoting process ,so one additional step has to be done that is just find out element of that column okay below that row which element in that column ok, corresponding the whole column is the largest element.

So I suggest so you programming the same example which you are using before, so now with the pivoting, because the same matrix which our using before ok, which has right hand side the 7.65, 13.5 and 16 and then we would, now do one most step in the elimination process see. So in this elimination process which I had okay, so we had okay, so we had if you remember that we just those element and then big row from k to k equal to 0 to k equal to n minus1 okay, and then eliminate for k equal to 0 eliminate all the elements in the first column below the first row and when k equal to 1 we eliminate all the elements in the second column below the second row etcetera but, now before we eliminate the first column okay, what will do is, we will find out what is the highest element in that column is okay, I am just using this, using this array called s which was array call s and I store all the elements of the now, I go through all the elements of that column, I go from i equal to k to i equal to n that is, all the elements of that column below the k th row okay, so k equal to 0 all the elements ok and, if k equal 1 it is 1to n and k is equal 2 and it is 2 to n etcetera.

So I will go through all the elements of the k th column okay ,and I store that in this array s and then I have this, that is I just print out that elements here and then I have program have function call max which will find out what its maximum value of the elements, of this elements which I given and which number it is given so store them in the array ok which elements array and which element in the array is the maximum is it find out that simple process, so I this is the this program, it does that for me right so this program to this program, I pass they number of elements in the array that is n by n matrix like this right if it is n by n matrix like this, I am taking the first column.

So I will have n elements in that, so that n elements that s array, s which is s array s, array will have a_{11} a_{21} of up to a_{11} and that I go throw that and where as find out, I will start with say that j max 0 and take the first element as the maximum element, assume the first element is the maximum j max is 0 and x max is, x of 0 that is what the issue then I will go through whole array that is s array okay, and if starting from 1, okay the first element taken has the maximum assume that and I look at the next element and if the next element is greater than the maximum value which I assume to be first element right now, and then I switch, I make that s max as that particular thing okay and I store j max as j.

So just I am looking that pair as look at the neighbor right so, start from 0 and loot at this neighbor if and the first element start from 90 and 0th element given the status of the maximum yea, and then I look at the neighbor and neighbor it have higher than itself ok then, the status is fast and to the neighbor the neighbor is now the maximum okay, and if it is not that its ignore the look at the next neighbor etcetera.

So whenever, it is fine some element maximum more higher than itself. Okay then, that was status of maximum its past and to that the element and that the number that array element the store in the j max ok and return to j max here is also want to use this to show you that how do how do right your function. So here is a function so, returns integer so just simply returns so what I am trying to say that I if I fast to this program whole array elements 0, 1, 2, 3 etcetera, it is look at which element or which row in this column is the highest returns that row element, the row number ok, so that is what the program is doing.

We just go through this okay, and it returns that row number okay, so it expands to show that is the way because it returns a integer, it this program, this function is also an integer that is type integer. Okay it is type has, it type is set of integer max, it receives the number of column elements, the elements in the, an the elements itself, we have put x of 3 the dimensional because of 3 by 3matrix I am doing it as example.

So it is return the value okay, so once return that value , so we go here and we have the j max, so see j max term of the grater than 0 ok so j max grater than 0 than have to interchange the rows okay that is what I am doing here okay, then what I do this store the all the elements in that column, okay all the elements in the row, so I know the j max is which to be my maximum elements, okay that is if j max is 1that means the particular column, in that particular column ok, the k th column, the k th column k plus 1th row is the maximum, so it is k_0 means the first element, the first element its maximum the interchanged to the two rows.

So what I do is, I store all less elements of that row into the same s matrix s array then, I transfer all this into that array and then I transfer the s it has store into this array that is what I am doing here. So I am storing, I have to switch between the k th row and k plus j th row, j max row okay, so j max not equal to 0 then, I have to switch between the k th row and k plus j max row right because k plus j max is the row containing in the highest element in that column. So I store the k th row element into the s okay and I put in the k plus j max row element into k and then I put k plus j max row element into s here. okay so, k plus j max elements have been put into s that is what I am doing here. Okay I switch this rows and then I similarly, know also switch has right side ok of this a problem we remember, okay when I switch this rows here okay, but when I switch this rows with these entire rows switch because this happen to the bigger than a $_{11}$, okay, I find out j max 1 so I have to switch between this and this row ok when I do that I changing also switch here b_1 and b_2.

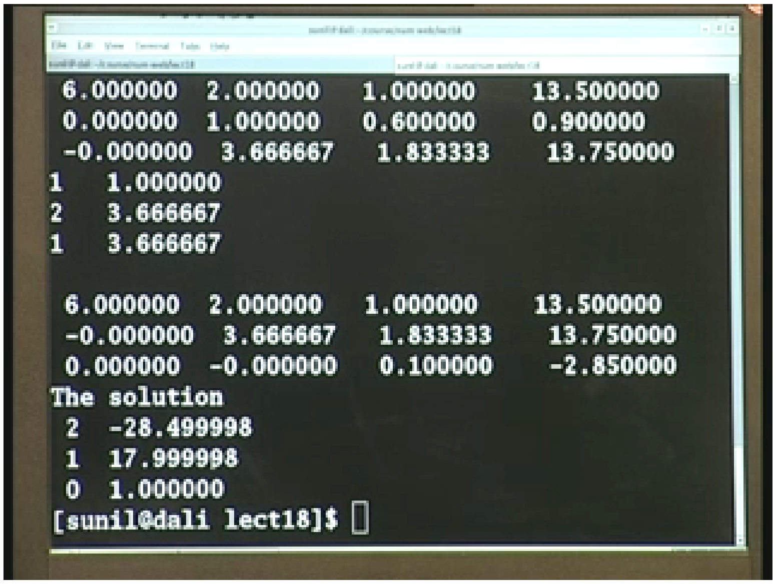
Okay otherwise, I would be getting wrong equation this side switch this also, so now that is what done here okay, I switch also show the d_k and d_k plus j max and d_k plus j max given a max what about our store in the, in this array s_t. So now, that is the switching and once the switching is done then, I go back and continue which is x call, I go back and once switching is done then I eliminate that column, okay and then go and continue and just show you the implementation of what happened when I run. Okay, so here is the initial matrix right, so 321.1601 and 10142 the initial matrix righted side 7.65, 13.16.

Okay so now, I looked at the first column, ok that is 361, so that is 012, ok up to this is. So 361 was my first column and now my first idea is to, for the first thing which I have to do is we eliminate it to the second row and the third row elements from the first column okay, that is what I have to do. Okay, so right but that is what I have to do. So look at the first column and I see at the and I look at the which is the maximum value that column it turns out to be the second row, okay distance start from 0 row number is 1, so row number is 1 the maximum number value 6.So now, I have to switch right so, I have to switch between this row this two rows and that is what I do. I will switch between this 2 rows make everything below that 0s that is what I have done okay.

So, I can see that second row is now, gone up first row ok so, and second row has been eliminated and third row has been eliminated, ok that is what I have to done. So I have switch this to 2 rows and then done the elimination process and you see I switch then also right hand side. So you can see from here, ok and then continue that I look at the next row again, I have to eliminate things below that okay, so I have 1and 3.6 here and again I have 3. 6 here because of the one right, so I can switch these two rows, I will switch to second and third row in the second step okay and then, eliminate the last row, okay that is what I could be doing here because the switch this if find out again that the 3.6 is the, this is the largest element, so I switch this to, ok then made as the second row okay then eliminated the last row ok, again you can see at here it is also switched okay. So that 13.75 it is here, it is gone up, it is gone here right.

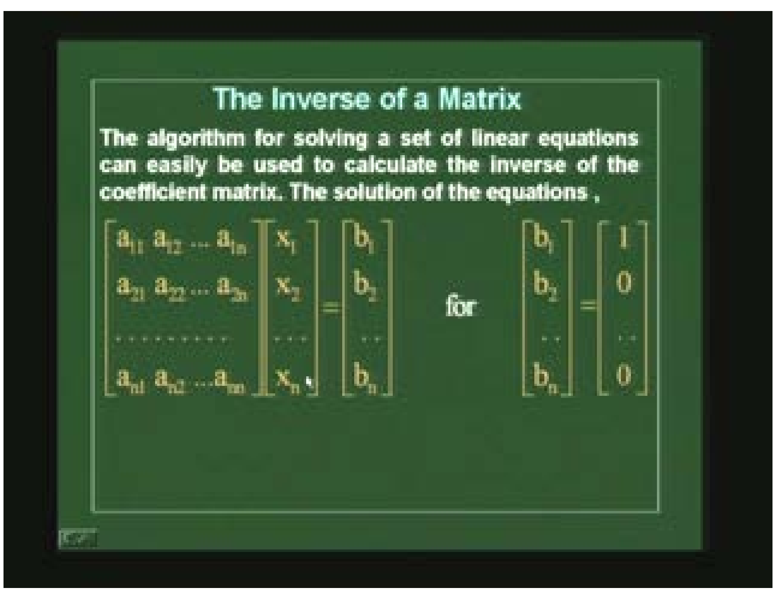
So that is the process which we see. So now, we have the upper triangular matrix, I can do the backward substitution, so the upper triangular matrix which right hand side again do the backward substitution okay, this is a simple gauss eliminations with the pivoting, okay and then I can see at the solution I get the exactly the same of as what I was getting before okay. So only thing here that is, consider more accurate because we always make sure that we obtain the highest accuracy which we can get by the program because by making you sure that the denominator of the factor of the largest element in that column, the idea of pivoting. Okay, now getting back to the LU decomposition and using that to, using the universal matrix. So, as I said to before, so we can write the LU decomposition things as, so we can write matrix a as LU decomposition like this right, once given the matrix we can write this LU decomposition now, idea is find out inverse of the matrix ok so that see, so universal matrix.

(Refer Slide Time: 23:47)



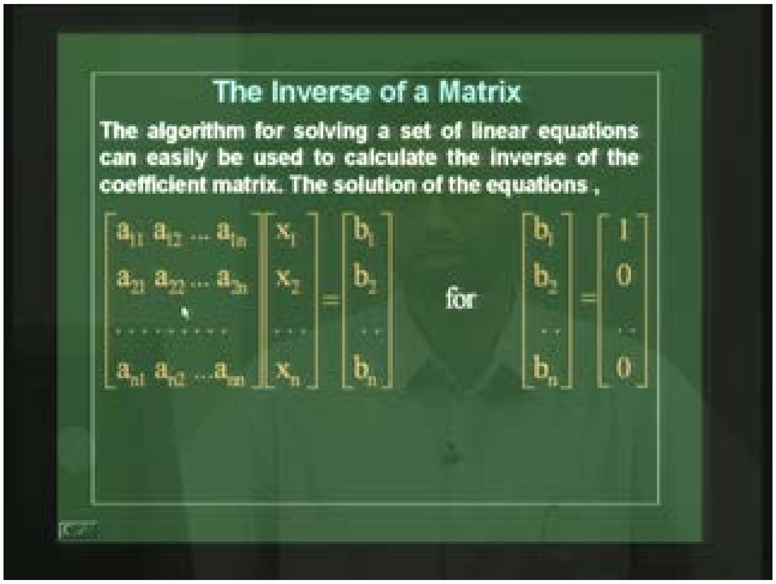
Now, how do we find out universal matrix, so way to find the inverse the is the following that, we will use this matrix a and then, we will substitute for b_1 b_2 b_3, we will substitute 100000 that gives as one solution, one set of solution x_1 x_2 …x_n, and then we use same matrix but now replace right hand side by 010,010000. So basically replace the right hand side by elements column of identity matrix right, so we will get and here that columns and solution here will be there the column here inverse matrix right you can see that very clearly this matrix multiplies by this column factor gives as, if this right hand side is the column of identity matrix that is 100000 and 0100 etcetera and then corresponding solution would be if the column of the inverse matrix

(Refer Slide Time: 24:19)



So columns of inverse matrix can be obtain by replacing this equation right hand side of this equation with the re column of the identity matrix ok that is, so what will be do okay then will find out the inverse matrix. So now, LU decomposition is especially useful here because, once we have found the LU decomposition form of these okay, we do not have the change which has have replace this columns okay this column the right hand side of the matrix with this column of the identity matrix that is what we have to do.

(Refer Slide Time: 25:39)



Okay and this matrix as same we will keep hearing the solution so we will just look at the implantation of the again. so this program which using LU decomposition to find out the inverse of the matrix, so again so we have matrix d, ok we will probably uses same okay and then we have at now this is which store of the right hand side of the matrix right and now, I have another so, I have another one which store the LU decomposition form is what I do its here also I have 2 matrices here I got actually 3 matrix that things, we do not need to have 3 matrices, 3 arrays but I just used 3 arrays for illustration. So we have this array d which is the 3 by 3 matrix and then have see also by 3 by 3 matrix we have s 3 by 3 matrix ok, and this y will be store those intermediate solutions which of the LY equal to d solutions because, we are using the LU decomposition form.

So we have to do matrix here, okay this matrix d okay, now I store the matrix to this columns see then the another matrix c into the another matrix c okay ,so and then so why are do this. Okay so I am, I have to this matrix d who is the inverse I am support to the find. So before do anything to d matrix okay I have store the elements of the d matrix into another matrix c, c and d same to start okay that is same matrix okay and then I will call this function okay, what are done is I have I want to find the elude composition of the matrix and I want to find out only one right and then I can use, I can change the right hand side of this matrix equation find the inverse, so just I need find out the one, so I just need the function which would find the LU decomposition form.

So that is a function, so I call this function d, d is now my 2 by 2 arrays okay, here is the program which passes a 2 by 2 array to a function okay, we have seen in the earlier the beginning when are a looking at the program part 3 different way is actually passing a 2 dimensional array to a function right we could pass 2 dimensional array to 2 dimensional array at 2 dimensional array it a_1 dimensional array a pointers or we could pass 2 dimensional array to a 2 dimensional array of pointers. So here, I choose to use the third method using this this function LU which takes and then d and the order of the matrix is n its two here that is 3 by 3 matrix 0 to 2 ,okay and now this function now, receives it as a 2 dimensional array of pointers okay, as this the function which received the 2 dimensional array of pointers and then it does the decomposition , okay now ,how does it do the decomposition we know, we already seen this, okay then see the decomposition by the elimination right.

So it uses the elements of that array okay and find the factor which it is multiply to get, to eliminate the column. ok that factor is here okay and then store that factor is a x for example here into the element itself, so after the gone through this particular program, okay it is actually takes this matrix d has 2 dimensional array ok and then, these the elude composition okay, so take the change go through again LU decomposition but the elimination we have the factors, for every column for every row, we have factor for every row each factor and then whatever the column elements has to be reduce to 0 is to use to store that factor, so that is what we have done, so this factor that is store in the particular d_{jk} element itself okay, for every k every j value okay which our gone to 0 the d_{jk} elements reduce to 0. Okay, that factor is used to store the particular factor that is our Alfa matrix.

So only one more thing, what we want to you to pay attention to and that is it 2 dimensional array of pointers ok, when use those elements we need to use type casted ok that is, what this it is okay this is, this is now using the two dimensional array of pointers that are getting using is are notice the way we use it if put in bracket with an accesses d, that is important. So that is have using because have it is c it has two dimensional array of pointers we could of received it has 2 dimensional array itself in that case, we do not need to here because and used it has d straight away. So now, see we pass the elements d into this particular program use that use that matrix and corres put into l and u matrix write and it store in the same matrix. So now, because as we know all matrices, when you pass by reference, so when I changed to inside of the program inside the function ,when I changed do matrix so ok that it alter in the d matrix main program also right.

So that is why need to store before I pass that into this, I need it store this element in another matrix c. Okay I store that now, after I call this LU function okay now, the d I have is in the LU decomposed form. So my original matrix now, stored in c ok which most of the cases may not need this I am storing is just to check and what I am greeting its actually inwards ok so I store that here. Okay otherwise, we do not need to store this, so I store that here just for, just check that what we got n design inverse matrix and then have a LU decomposition program, this program is call and return to me d, which is in the L U decomposed form, ok now that is the structure. Okay now, just is stored is two loop its store it and then, we have the LU decomposition now, we just print this LU decomposition matrix here. So now here look at the first.

So we print and has the store original matrix ok 321.16 0621142 that is our original matrix right and then, we have the LU decomposed matrix okay, so now of course first does in now change in the LU decomposite matrix ok and I want to use pivoting here. Okay this first does in change and this my LU decomposite matrix and, then I have okay once we have in the LU decomposition ok just show is that ok that is upper triangular matrix we can take it as 321.1 and then minus 2 minus 1.2 and minus .366667 upper triangular matrix the lower triangular matrix then would be 12.333 and 1 here minus 1.6667 this will be, this will be a lower triangular matrix and one here then be lower triangular matrix okay.

So now up to that we have to find one you have to LU decomposite matrix we can go head find the inverse matrix so the matrix next part of the program just find out the inverse of the matrix ok. So now see what I do is I have 3 column is right is a 3 by 3 matrix its m goes to 0 to 2, okay now there is m is m, so some integer and go from 0 to 2, okay that is my 0 in my 1 and 2 okay the first time when m is 0 it the right hand side of the matrix okay is I initialize every thing is to 0 and right hand side of the matrix the d 0 is 1 every thing is 0 okay every thing is 0 the right hand side okay d_0 is 1 and then I find solution using for substitution here in the l matrix and backward substitution the u matrix and I find a solution okay.

Now that solution I store has of first column of the matrix column of the matrix called s. Okay, so what I got solution have with d_1 has d_0 as 1 every thing l is 0 is stored as the first column see m is 0 here right, the first 0 column that is, first column the of the of the matrix “s, i, m” of the matrix x. So every the solution goes to here this is backward substitution of u, UX equal to y equation, ok the store is the first column I hope this is clear that is that we have the m going from 0 to 2 okay and the first time ,we take in I put in the first element as 1 ok that is m is equal to 01 and everything 0 and I do the backward substitution to get LY equal to d solution there is y is obtain from LY equal to d for substitution an then I do the backwards substitution in the get UY equal UX equal to y, so and I get the s there is the solution every thing and I store the that is the first column that is m equal to 0 from different the first column of the of the matrix and then go back will here okay and an change equal to m equal to 1 okay and again m equal to one everything its 0 accept to d1 so there is 010 my d matrix right hand side and I repeat the same process and end of once have gone throw all the way up to n in this form all the way to 2 there is 012, I have the full in s have the inverse of the matrix ok that is what a stores.

So once I do there inverse of the matrix, so I have inverse matrix. So once have the inverse matrix, I can as use check, by multiplying this matrix that, I get the product has the 100 is the, this product matrix that 000 element is101element is 0, to element 010 element is 001, element is 112 element is 020, element is 021 element is 0, and 22 element is 1.So I getting the identity matrix and the right hand side, so now another way to, checking this would be in to checking to go back with this as my original matrix find the inverse right as way checking the inverse, the inverse should be same matrix started with, that also tales how bad is the errors in the program, okay because with round of its extra how bad they are, they are seen by doing by the this inverse of this inverse, so when you try to do limited see the problem here the first element here is actually 0.

How the inverse matrix, so without pay outing he will not able do this inverse of this matrix, so here is an example of k is you for need pivoting. So you need to interchange this 2 row before we proceed. So but, that also brings into one more point why have not using pivoting here, ok show do you to use pivoting here and why are use pivoting here the reason as the following here be doing LU decomposition ok and why do LU decomposition will be changed the right hand side of the matrix you only change in the so LU decomposition only changing this matrix we only change this matrix to upper triangular and lower triangular, okay I do not change this right hand side, okay because I use the full LU into x equal to b the solution equation, I do not want to need to change but when I do pivoting I interchange rows, so then I need to keep track of what happens.

Okay, I need a permutation matrix and that has the multiply by b, b with multiply b with later ok otherwise, you get the wrong answers, here using LU decomposition with pivoting we will standard program that apart from the LU decomposition, if you are using function which at is LU decomposition for you with pivot returns to the original matrix in LU decompose form the upper triangular and lower triangular the matrices pivot the matrix is the same okay is also return a permutation matrix which you need to use to ha to interchange this rows on this b means we need to some of the store the card of what exchange how rows you maid here and do as the exactly same thing on the right hand side ok that is something not implemented here because little more involve and we will do that and as later stage of the program. So now, we know that how do inverse of the matrix using an LU decompose form.

So now that we will discuss a little bit more on the possible errors which it is, which can in this kind of calculations. So one way to quantify such errors would be this is to look at, what is the reminder which we take x and subtract from b and we solve this equation a_x equal to b. Okay and then, you do the you are the solution finally another the value of the left hand side from the subtract the right hand side and if ever thing is correct then should r equal should be 0, so we will find r is now 0 because of, the round of error etcetera coming in this process.

We will also have some small error received on the left hand side how this part. Now, what kind of value switch can dollar rate as the request when ,so on accuse the need for some easily but then 1 this 1 by more quantifying that and that is following now, if you get r as a, a into x minus, a into x minus, b minus ax that is r right but if you solution if x its exact solution then we have to value 0 right okay, so the exact solution is equal to x with bar, x bar and x is the exact solution then that received should be 0, ok then a_x bar.

So than I can subtract r equal to b minus a_x, r and 0 equal to b minus a_x bar and right sequence of this. Then, we will find that, whatever the deviation of your solution from the exact solution multiply by a easily is your residue. Okay, so depending upon the a matrix uses, even its possible that even small deviation of the solution from the exact solution can give large reseeding here, which an large then a matrix or the deviation of the solution is equal to a inverse of the reseeding right, what about means if the, find that look at find a solution for x, a_x equal to be a solution and residue, it is small may my solution its required good quit good and senserved its very closed actually solution because, my reseeding is a says small but right it is this form ,x bar minus x is a inverse of r.

You find that is a inverse small, a inverse is large, if some the elements of a is inverse large that say and then m may reserved is very small the solution is still very bad because the actual difference is a inverse is large, is a inverse is large matrix. So, therefore the finding the a inverse is a sometimes is very necessary have to calculate to even the fine and the solutions of the set up linear equations by the elimination process the what our they makes.

So now, this can be quantified on other way is, so otherwise first of all we can check the a inverse, so one way to reduce a inverse, so that is actually here. So a inverse should not be large, that means we should adjust a such that a inverse is not large, when we before, we attempt to solve this equation a $_x$ equal to b. One way to do that is just simply scale the coefficient of the a matrix, the a matrix itself scale we know that if just multiply the whole row of the a matrix are the both right hand left hand side of the linear equations, by accounts and factor others is not going to solutions, okay so but improve the accuracy of our solution.

So these are some numerical tricks that we should remember, when we solve a simple thing in a set of linear equations that is, we could have large errors because the round of and that getting amplified because of this reason one way to make sure about that always reasonable number okay so for that which can do actually scale the a matrix that is that the largest element any row is one, and then we calculate a inverse. Now, even after doing that any of the elements inverse matrix still large than 1, much larger than 1, then we much other can do above what was and we call that system has ill condition is that is, say definition what is call a condition, so if the a inverse still very large after scaling of the elements, we still get scaling such that large this element in any row is 1 even then a inverse is large value and then, the whole equation is called ill conditioned equation.

So now another way to find out the condition of the matrix would be as I said to a into a inverse, if the this not equal to1, is not close to1, then we will again say the system is ill conditions, in that sense program is looked at the good conditions because the we did not get the reasonable good values as a into a inverse. Okay so, now we could always said in the earlier case, so the third process would be that we will actually find the inverse of the inverse right and then compare with the original matrix.

So another way to finding the condition of the matrix, so that 3 different ways find in the conditions as see we could either level elements and then look at the inverse of the inverse is the large then, it is ill conditioned or we could be multiply matrix by its inverse and then inverse of obtain numerically and then, see how good that is how close to unity for identify matrix, it is and if it is not closed identity matrix, then it may system its ill condition are the third way to used to actually find the inverse matrix the find the inverse again ,ok and that is not closed the matrix we started with again do say the matrix is ill conditioned for this kind of process .

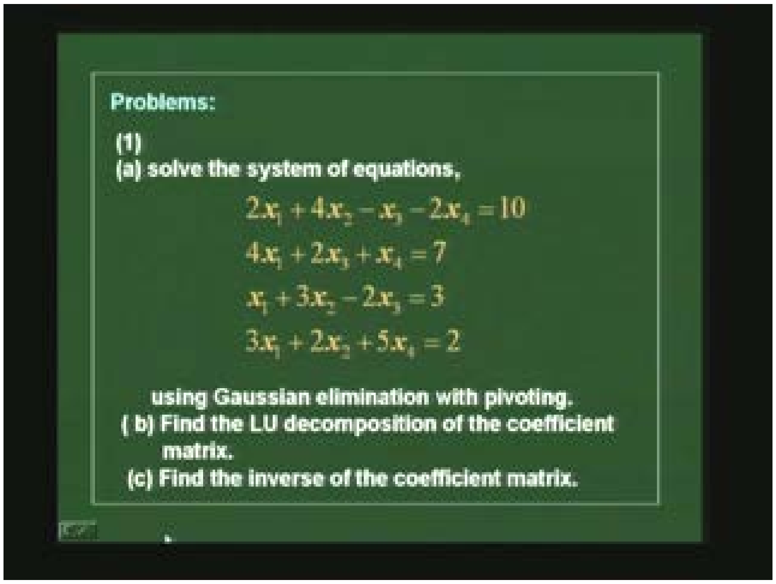
Okay so now, we will that condition concept of the condition will now quantified and will count with the number and that is call condition number. So that is using the concept which we just talked about, ok that is condition number of the matrix a, is a time is call norm of the matrix a multiply by the norm of the matrix a inverse. Okay now, here is one find one way of finding the norm again that is does it nation of norm, I do not go into the details is the analysis here okay, just tell you what this is that is one way of the finding the norm is a just the sum all the elements, all the elements of the matrix, sum of the square of the all the elements and the sum square all the elements the matrix is one norm okay, this is something which is invariant and the rotation of the matrix etcetera. So this is the some of the square of the element of the matrix, ok and that is called a norm then equal to compute this, the norm for a and its inwards and take the product ok that is call a condition now, that is they are quantifying the condition. So we will see the use of that, ok so now here it is what the significance the number right.

So the reliability of the solution of the equation, so the form a_x equal to b right, so way that given by so that is what you we are interesting looking at right so we can see that as delta x by x given by condition a into delta a by a that is a reliability of solution. So delta x by x is the deviation from the actual solution okay, we got in x and delta x by x, x is the relative have right in the x, so that is its actually equal to the delta a by a .Okay, is it relative into condition now, the relative error in the norm of the coefficients of the matrix again I have that is the, that the connection, okay so delta a by a is the relative error in the norm of the coefficient of a. Okay now, this is how mind, okay that we have all the ideas having a condition number, okay we know about compute the inverse I can since, so we can compute inverse the condition number and we can use the condition number it is a quantified, the errors you have, we could try to solve some problem and then, we actually check whether what is the reliability of the solution is in a practical way.

Ok now, here is a problem which we should be looking at, ok here is set of equations, ok which here is solve no which as increase a elements 0 that is the z as the first row of my matrix would be 2, 4 minus 1, minus 2 and second row now is 4, 0, 2, 1 and third row is 1, 2, 0 minus, 1, 2, minus 2 and 0 and fourth row is 1, 2, 0 and 5and right hand side is 10, 7, 3, 2. So we could try to do the LU decomposition of that and, we could also Gaussian elimination with pivoting and LU decomposition without pivoting. We have the program for that, so we could do that Gaussian elimination with pivoting LU decomposition without pivoting and we could find the inverse, once we have LU decomposition form, we could find the inverse of the coefficient matrix and then we could check, what the errors, how do, how do this value would compare, the solution of compare and that is something which use should try do it.

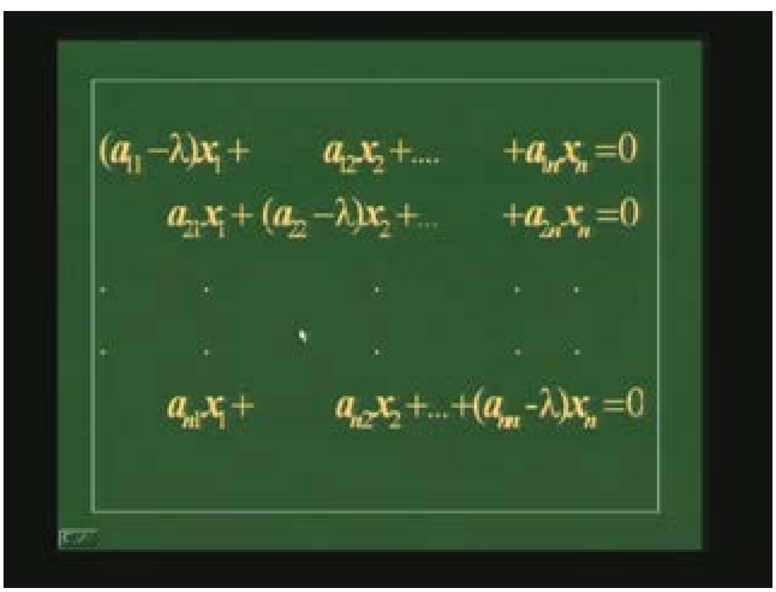
So for we look at the set of linear equations and we used the matrix, so matrix method to solve the set of linear equation. So we continue a little bit more on this matrix equations and look at the some other property of matrices for example, we look at, how to compute eigenvalues of this this matrix is form for example we solve the matrix equation of the form a_x equal to b that is what be just learn. Okay now, we want to look at the cases for b equal to 0, okay the right hand side is the 0, okay so in that case this equation a is equal to 0. We can find the solution by that, by looking at the inverse of the matrix a_i. So if b is non-zero, we could compute the inverse of the matrix and then multiply both side by the inverse solution of by a.

(Refer Slide Time: 50:15)



So we can find the solution of the this by looking the inverse the matrix but we could still you know find some non-trivial cases but eigen values for example, okay one of the example we have to solve equations of the form a, a_x is equal to 0. So it is not that "b" is always non zero in the set up linear equation, we are interested in he always cases for b is not zero that is what true, okay that will case were b is 0 that is one example of that is the finding the eigen values of the matrix, so we will see that, now that is we will what something which we look at. So for example, we want to look at the Eigen value of the matrix a, okay then we know the characteristic which we would write is something like this a minus lambda x is equal 0 lambda i equal to 0.

(Refer Slide Time: 52:34)



So that is, so that is this form, we may have to solve matrices of this form, set of linear equation this for the right hand side is completely 0 right, so something achieve you would now okay, so is this again equation of the form a_x equal to 0 right, a $_x$ equal to b right hand side is now 0. Okay that kind of so how to solve equations of this form that is what we should be now looking here. Our standard methods which look at so for ok will not work in this scale right. So what be just know of the matrix, so this form that is a minus lambda I_x equal to 0 right that is matrix a is equal to 0. So some matrix into x equal to 0 that what to going to interested solve it but you have interested in here is actually finding this values of the lambda ok know the actually finding the solution, so we would be more interesting in solving the equations the lambda right.

So now, if you have the equation of the form and you would thing that okay with this solution of the true l, because x equal to 0, if you have n equation this form x equal to 0 okay there is b right hand side is 0 we would thing right solution that a_x is 0 but x is 0 its solution but that manual is true for the matrix the determined itself is 0, okay a and x is equal to 0 and I said just that finding a inverse of the expense because is I multiply by the inverse in the this side I am get x but here will get 0, I get only trivial solution x equal to 0 only considered the case of the inverse then add x okay right or that the determinant you know the inverse of the matrix just joined to by the determinant but the determinant itself is 0.

Okay that is good find inverse the a matrix in that case we could non trivial solution for the equation right ok that is something which is be could interesting looking at so the case at the determinant is 0 and that what to be use to find the eigen value. So our characteristic equation for the eigen value a minus lambda i into x equal to 0, so a minus lambda that the determinant, a minus lambda i equal to solve for can eigenvalue cases that is a another form of the matrix equation is your interested for just solving a_x is equal to b, we might be in here situation, we were to find out eigen value for the matrix that is, we want to find cases that be determinant of these is 0, so then we can write the determinant and then solve for lambda and that is the right hand determinant, we have polynomial of this form in lambda.

Okay, so this in your right determinant of the matrix equal to 0, we have the polynomial this form and what we are find this solution for the linear equations it is, we have to show that only for ordered two but this will be ordered n. So we have to find solution of this equation before, we find out going solution of the this equations, first we have to construct this polynomial right. So summaries we want look at is cases where you have the equations this form that as we have given a matrix is a I want to find out eigen value its matrix. So then we have some a equations is a minus lambda I determinant equal to 0 right, we want to solve this polynomial.

So that is full problem but the first step to that would be to the construct the polynomial that is the coefficient of the polynomial as we constructed. What is the p_n minus 1 p_1, p_n have to p_{0s} p_0 to p_n minus 1 we need to construct, as that itself is non-trivial problem, when it comes to large matrixes. so what we discuss now is methods of constructing have this polynomial and then finding the solution of this polynomial then, once we have the solutions the polynomial that can three have the eigenvalue of this matrix a which has the lambda is called eigen values and then, for each an eigen values, each of this lambda there is a value x which would be, which would call the column act x is the function of the matrix we have interest of the look at this eigen values and eigen function that is what we would be looking at in the coming few lectures