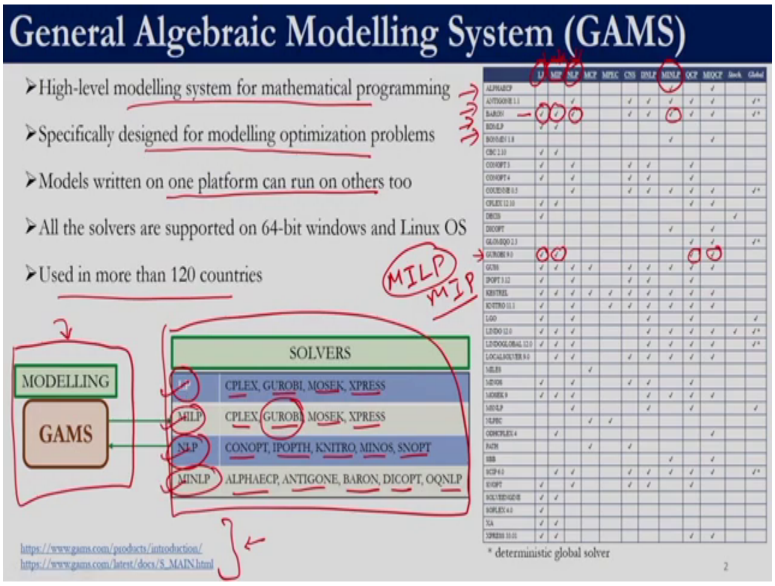
Computer Aided Applied Single Objective Optimization
Dr. Prakash Kotecha
Department of Chemical Engineering
Indian Institute of Technology, Guwahati

Lecture – 34
Generalized Algebraic Modelling System

Welcome. In this session, we will look into General Algebraic Modelling System GAMS. So, previously we had seen that, there is no inbuilt function in MATLAB which can solve MINLP problem, a mixed integer non-linear programming problem which has equality constraints. But GAMS can also solve problems, which are non-linear as well as they can have equality inequality constraints and some of the variables or all of the variables can be integer. GAMS is a high level modelling system for mathematical programming.

(Refer Slide Time: 00:56)



It has been designed for modelling optimization problem. So, here if we see, we have GAMS over here right and then we have solvers over here right. So, in this session, we will be looking into only GAMS right; depending upon the license GAMS can access the solvers. So, as you can see here GAMS is a modelling platform right. So, this modelling platform helps us to write the models, which are similar to the algebraic equations which we will develop using pen and paper and then it has variety of solvers right.

So, for example, here we have shown solvers for linear programming mixed integer linear programming, non-linear programming and mixed integer non-linear programming. So, all of these solvers can be used to solve a linear programming problem right. So, similarly for MILP, we can use CPLEX, GUROBI, MOSEK, XPRESS. For non-linear programming, we have CONOPT, IPOPTH, KNITRO, MINOS, SNOPT. Similarly, we have for mixed integer non-linear programming BARON, ALPHAECP, DICOPT and other solvers right.

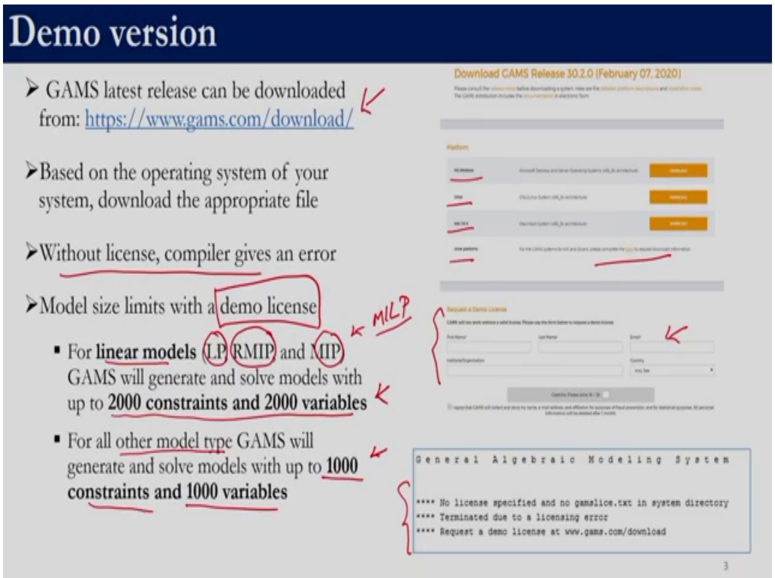
So, when we have to purchase GAMS, there is a cost involved for getting the modelling platform GAMS. And depending upon the set of solvers which we choose, we will have to pay to get those solvers; some of these solvers may be available free of cost for example, GUROBI is available free of cost.

So, the models which we write in GAMS would be compatible across platforms. So, a code which has been developed on windows would also run in Linux provided, we have GAMS on both of them right. So, GAMS is used in more than 120 countries and further details about GAMS can be found on their website. So, this table shows the various solvers. So, these are the solvers, the rows are the name of the solvers and the columns are the types of problem right.

So, in GAMS mixed integer linear programming is actually called as Mixed Integer Programming right. So, MIP actually indicates what we have been using is MILP. So, LP, MIP, NLP and MINLP, there are various other classes of problems you can look into that. So, now, if we look at this GUROBI version 9 it can solve linear programming; it can solve mixed integer linear programming. And it can solve QCP quadratic programming and mixed integer constraint quadratic programming.

So, if you look into BARON, it can solve LP, MIP, NLP as well as MINLP and it also solves various other classes of optimization problems. If we have to solve a mixed integer non-linear programming problem, then it is not sufficient to just have GAMS. We also need to have an appropriate solver which can solve mixed integer non-linear programming problems.

(Refer Slide Time: 03:44)

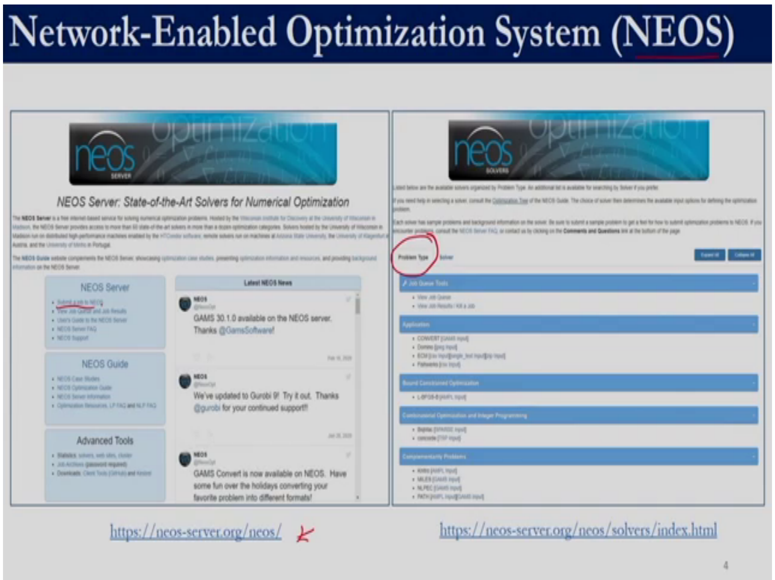


So, you can download the demo version of GAMS from their website right. So, here you can download it for Windows, Linux, Mac and for other platforms you will have to fill a form to request the corresponding version right. So, without license if we execute a GAMS file, we would get an error right. So, this is the error which we would get if we execute GAMS file without having the appropriate license.

With the demo version we can solve linear models right. So, when we say linear models it is linear programming a mixed integer programming or MILP. So, remember as we discussed earlier, mixed integer linear programming is termed as MIP in GAMS and this is relax mixed integer programming problems. So, for linear models it can support up to 2000 constraints and 2000 variables for all other model types, it can support only 1000 constraints and 1000 variables with a demo license.

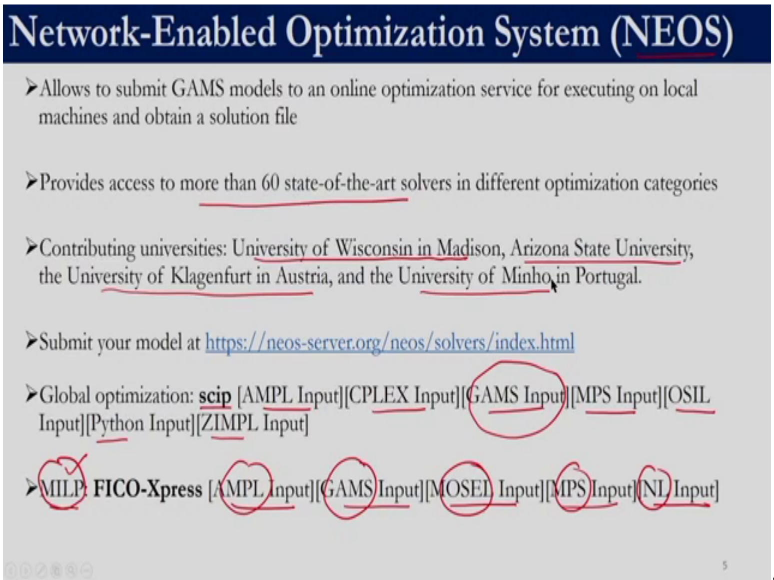
So, this demo license can be obtained without any cost right. So, we just need to fill a basic details over here and we would get a license file to the email address which we provide here that can be used to install GAMS on your system. So, when you install GAMS on your system, you will get two option one is ida option and another one is GAMS studio option. So, for the rest of the discussion in this course we will be working with ida option, you can also install studio option and look at GAMS health which is available to continue using the studio version.

(Refer Slide Time: 05:16)



So, the bottleneck which we have over here that can solve only problems of a specified number of variables and constraints can be overcome using NEOS. So, NEOS stands for Network Enabled Optimization System right. So, this is the website for NEOS server. Once we code our problem using GAMS right then we can submit our job to NEOS right. So, this is an online solver and we will have to select the type of problem, which we have and if we submit our job online we can get the results over email or it is displayed after the completion of the problem.

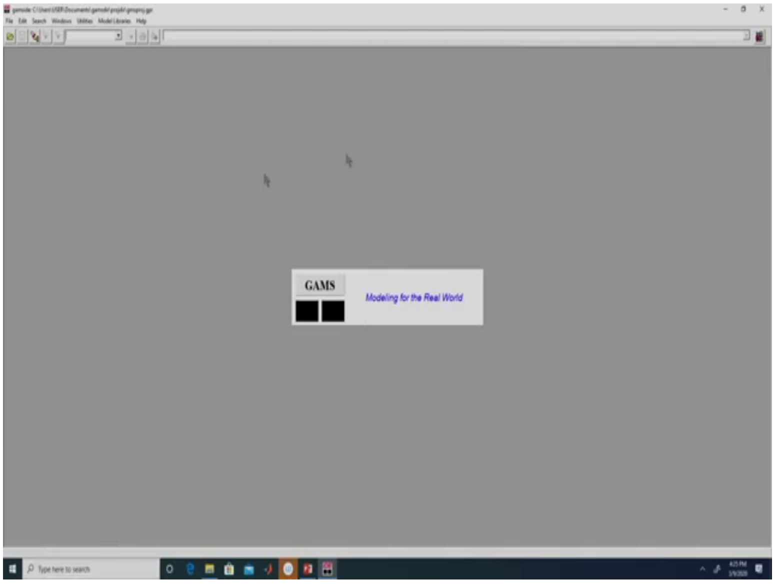
(Refer Slide Time: 05:43)



So, NEOS not only supports GAMS file, but it also supports various other types of input. So, for example, if we take this solver scip, then we can also input AMPL file a CPLEX file. We can also give the input in terms of GAMS as in terms of MPS in terms of OSIL we can also give python input and ZIMPL input. So, we will restrict our discussion to GAMS input right. So, for example, in MILP we can given a AMPL input a GAMS input, MOSEL input, as well as an NL input and MPS input. So, these are various modelling platform or it is a type of file that can be given as input, if we select a MILP solver in NEOS.

So, you can visit the website of NEOS to see the list of solvers that they support, currently they support more than 60 state of the art solvers right. So, the contributing universities for NEOS include university of Wisconsin in Madison, Arizona State University and University of Klagenfurt in Austria and the University of Minho in Portugal.

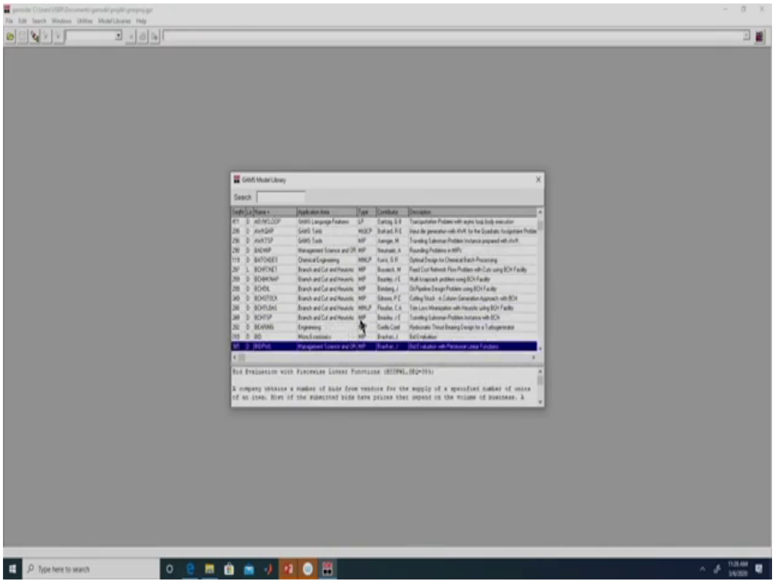
(Refer Slide Time: 06:49)



So, this is the GAMS interface right. So, over here we have these model libraries; in model libraries you will be able to find lot of problems right.

So, for example, let us click on this GAMS model library right.

(Refer Slide Time: 07:03)



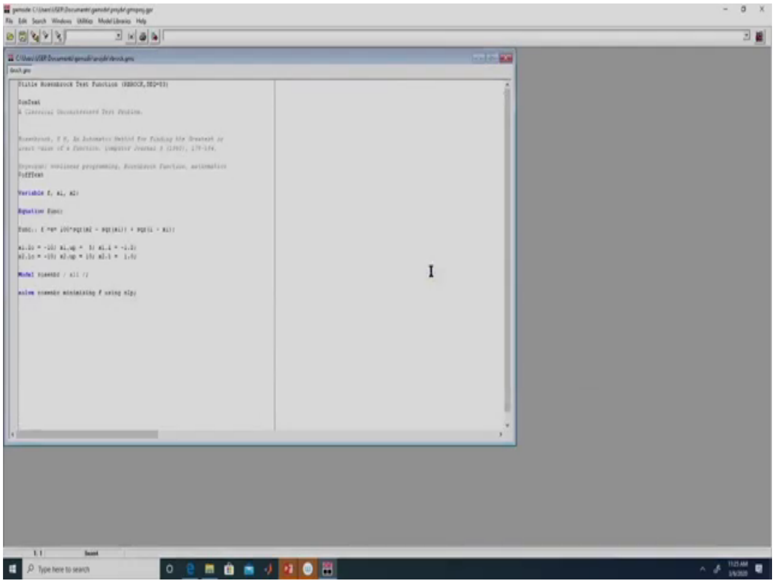
So, over here we have name of the problem right and the application area. So, for example, this problem Abel is from macro economics and the type of problem is a Non-Linear Programming problem; so, NLP. This column shows the name of the contributor and the last column gives a brief description about the problem. So, here it is a linear quadratic control problem right.

So, you get some information about the problem over here as well right. So, you can click on any of this columns right to arrange according to that column right. So, for example, if you click on this application area so, all the problems are listed in alphabetical order. We have applied general equilibrium chemical engineering problem, disjunctive problems, problems from energy economics, then we have engineering problems, finance problem. And these problems basically show some of the advance features of GAMS language right.

So, these are set of problems which are from management science and operation research right. These are some problems from mathematics and then we have micro economics statistics and stochastic programming right. So, we can also arrange the problem according to its type right. So, these are the various types of optimization problem. So, here we have linear programming problem LP; we have mixed integer non-linear programming problem MINLP right. So, in GAMS, they do not use the term MILP right.

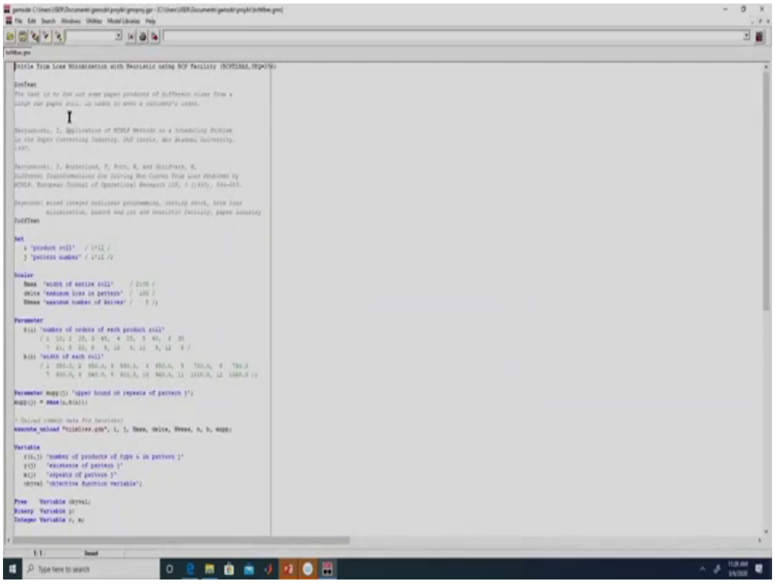
So, they do not use mixed integer linear programming, but they use the term mixed integer programming right. So, what we have been discussing as MILP in GAMS terminology they use MIP. So, over here we have that travelling salesman problem, which we had discussed in the introductory lecture right. So, this is a problem related to investment planning in the Korean Oil petro industry. This problem is a mixed integer linear programming problem and deals with portfolio optimization for electric utilities right.

(Refer Slide Time: 08:59)



So, these are examples of non-linear programming problems, this is the Rosenberg function which we had discussed previously. So, you can also sort them according to the description so, for any of this problem if we double click it right.

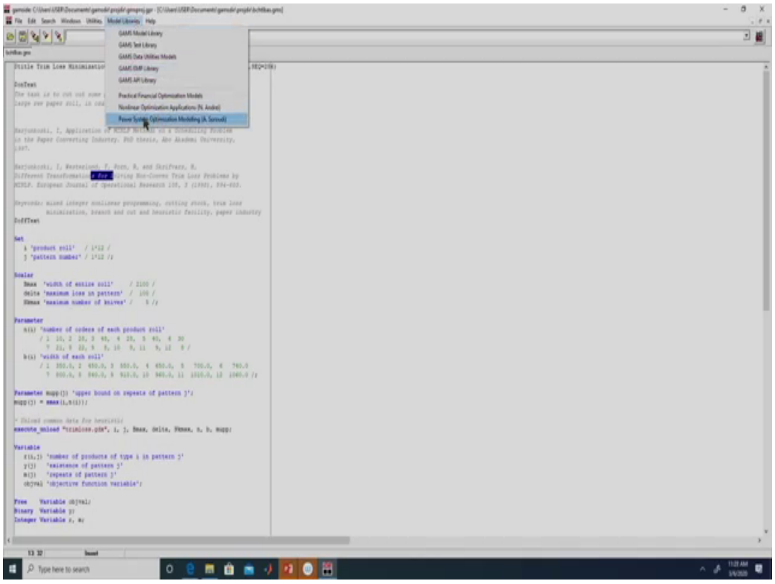
(Refer Slide Time: 09:08)



So, it will open the file one of the interesting feature about these examples is that it also gives us the source of the problem right. So, typically it is a book or it is a journal paper right. So, whatever problem is discussed over here actually comes from this work right which was published in European Journal of Operational Research.

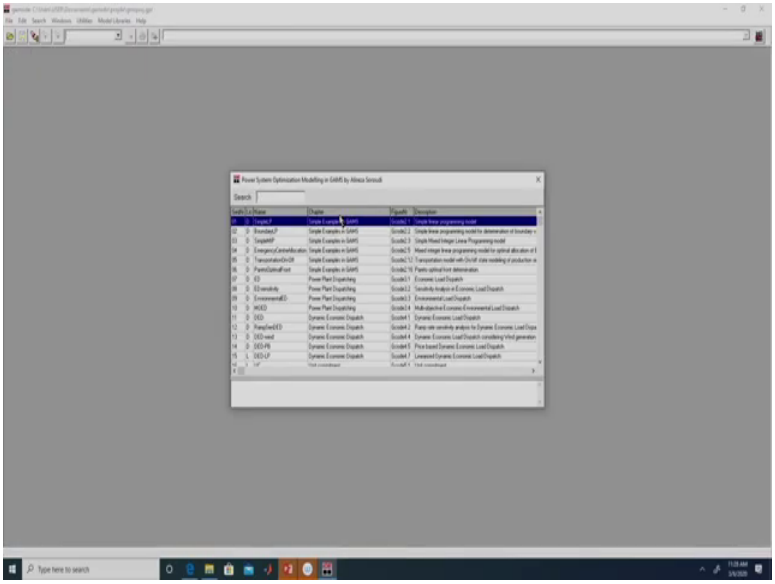
So, not only we get a code, but also we get a source wherein the description of the problem is given. So, now, we have the problem description as well as code right. So, if we are interested in this particular problem and we want to solve it with different set of values.

(Refer Slide Time: 09:43)



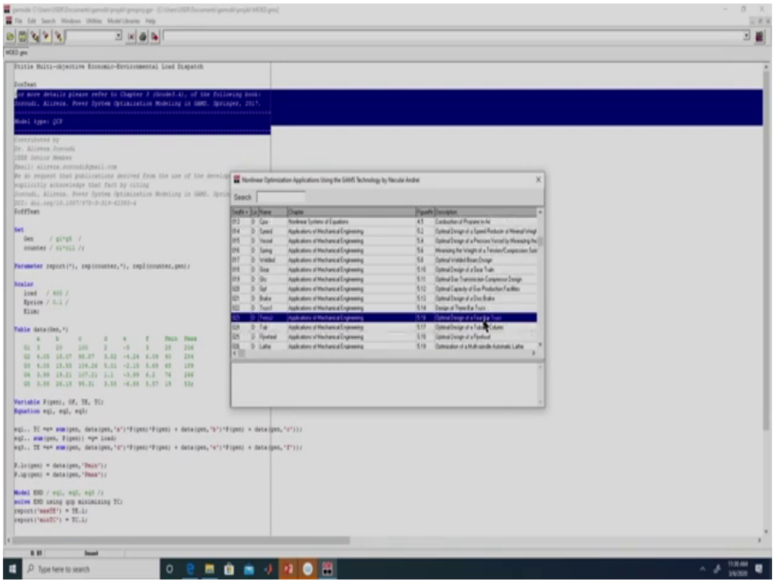
So, we can change the values and still execute this problem to get the results without putting in additional effort to actually code the problem; these are model library. So, there are some special model libraries also, this model entirely deals with power system optimization modelling right; so, from the book by the author Sorody.

(Refer Slide Time: 10:02)



So, over here the name of the chapter is given, the model name is given and also a brief description is given. So, for example, this problem is a multi objective economic environmental load dispatch right.

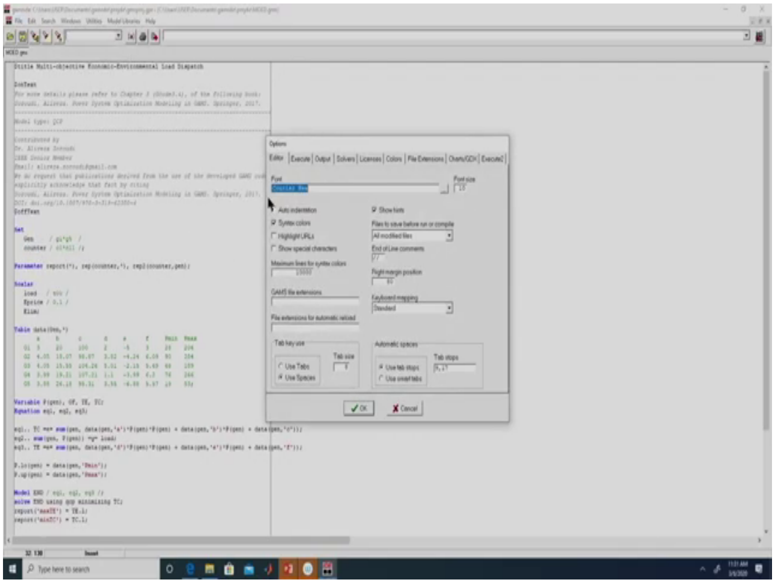
(Refer Slide Time: 10:14)



So, if we double click on that, we can again have access to the file right. And description about this problem can be obtain from this particular source. We also have a package on non-linear optimization applications right by Andrei.

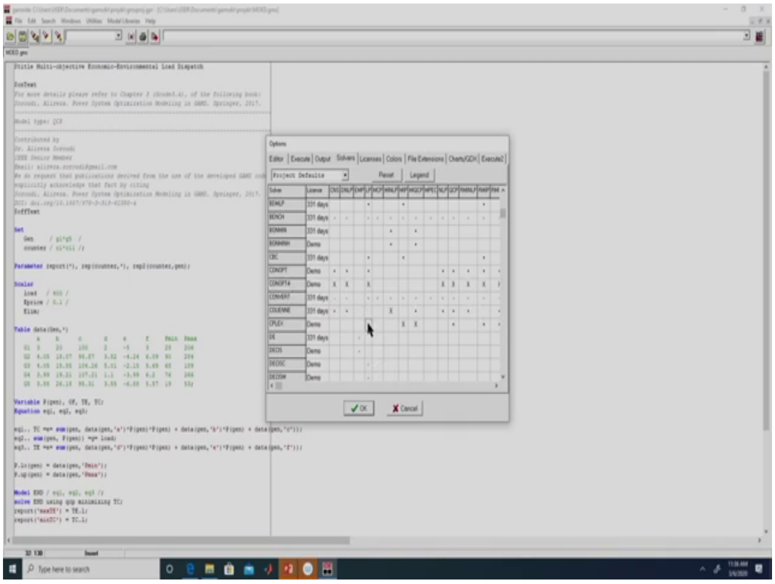
So, over here you will be able to find examples for a lot of non-linear programming problems right. So, for example, here we have a three bar truss design problem here, it is a 4 bar truss designing problem designing of a flywheel and so on. So, we can also change the default solvers in file we can go and select for this options right.

(Refer Slide Time: 10:48)



So, when we select these options a lot of things can be changed right for example, the font size in GAMS editor file. They expect you to explore all of this right.

(Refer Slide Time: 10:57)



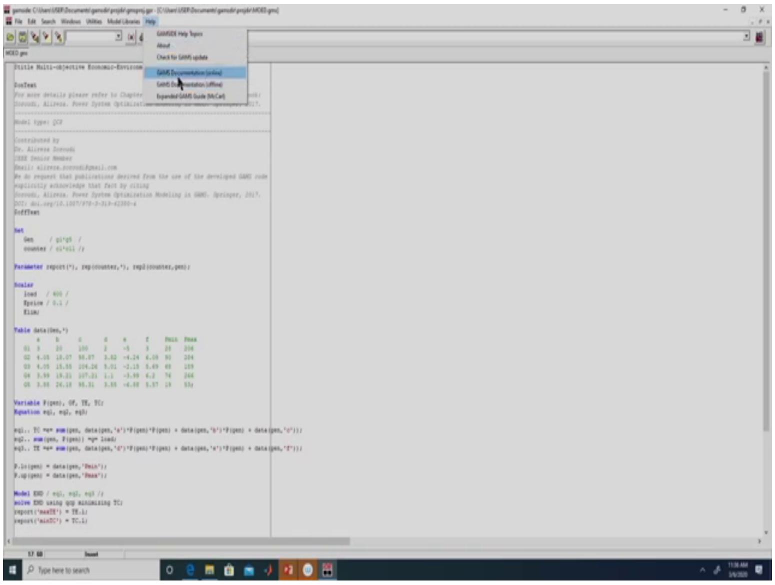
So, we will only talk about these solvers.

So, in solvers over here you will be able to find the name of the solver. So, this ALPHAECP, AMPL, ANTIGONE, BARON, BDMLP; all these are names of different solvers right. So, not all solvers are capable of solving every type of problem. So, over here ALPHAECP can be used to solve MINLP problems and MIQCP problems right. So, if we look at these solvers right. So, we have CPLEX over here right.

So, CPLEX currently it is in demo license right. So, it can solve linear programming problem, it can solve mixed integer programming problem it can solve MIQCP it can solve QCP it can solve RMIP it can solve RMIQCP right. For linear programming if we want to select let us say BARON solver right. So, we need to click over here. So, now, whenever we solve a linear programming problem, it will use BARON to solve this unless we override the options in the GAMS file itself right. So, similarly we can select BARON for solving MINLP problem.

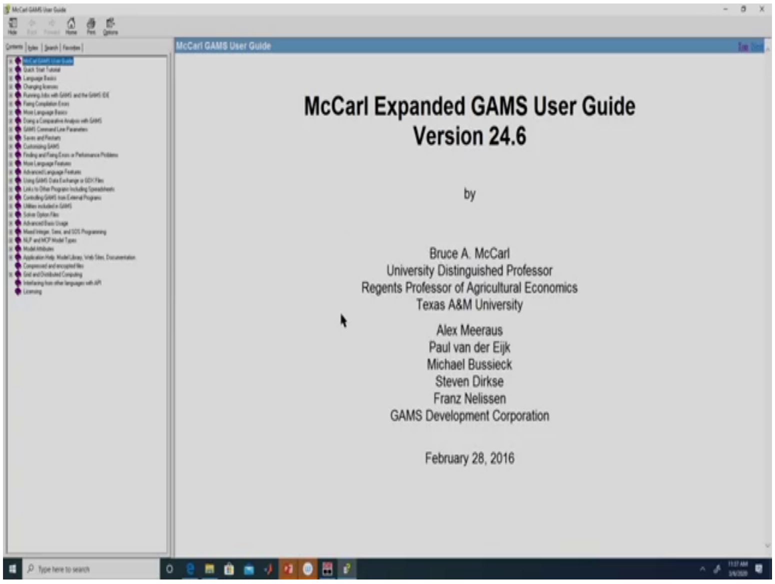
Right now Bonmin is being used right, but if we want to override that option, we can click over here right. So, now, linear programming as well as non-linear programming will be solved by BARON right. So, we can reset this to the default values. So, clicking on a particular solver name will make it the default solver for the compatible set of problems right. So, for example, if we click on CONOPT, then all this dots would become X right. But if we click on this particular dot right then only for linear programming problem CPLEX would be used right. So, for rest of the problems still CONOPT 4 would be used.

(Refer Slide Time: 12:37)



So, you can also look at this help right.

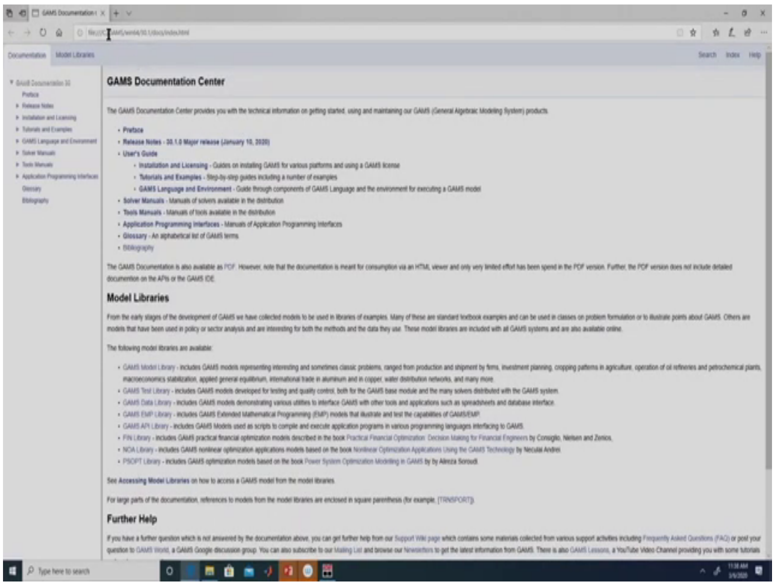
(Refer Slide Time: 12:48)



So, over here you will find the online help as well as the offline help of GAMS documentation you can also look into this expanded GAMS guide by McCarl. So, over here you will also be able to find some of the advanced features of GAMS.

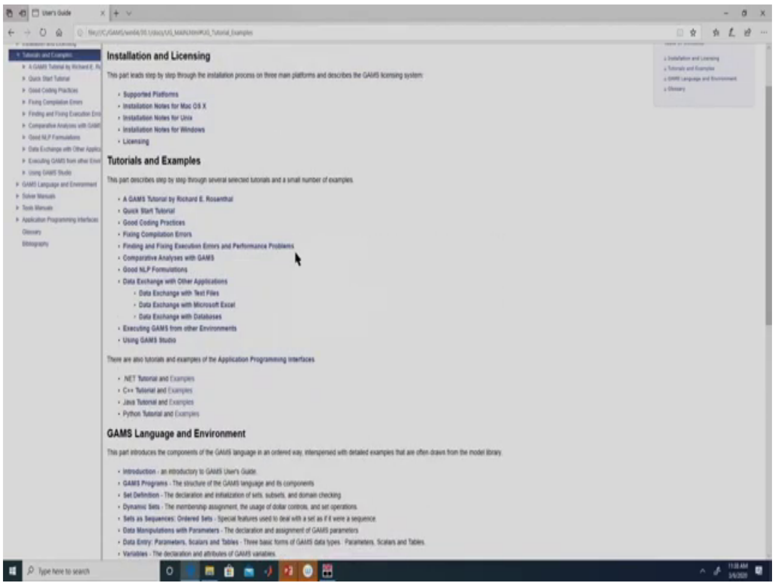
The example that we are going to discuss is from the offline documentation.

(Refer Slide Time: 12:57)



So, this is the offline documentation which is available.

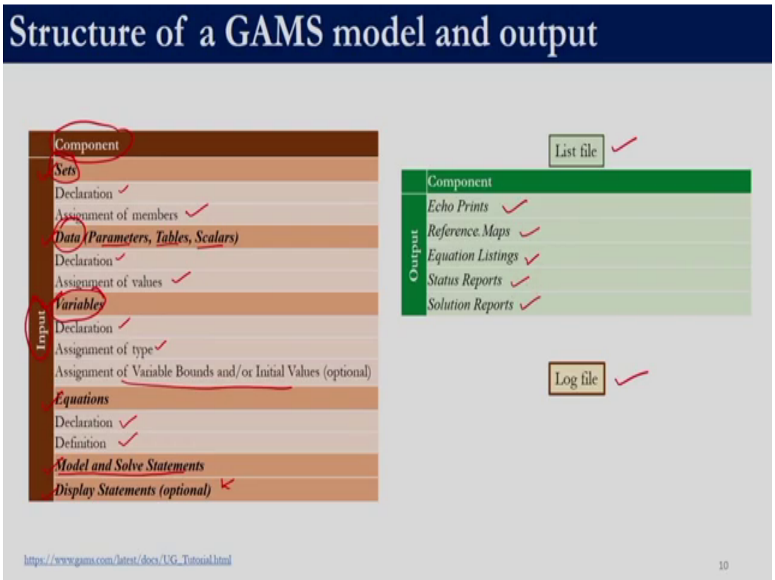
(Refer Slide Time: 13:00)



In the tutorials and examples, we look into this tutorial by Richard E Rosenthal. So, as in when you start using GAMS, you will require lot of different features right detailed help is available over here. So, you can look into the appropriate section of the help.

So, now we will look into the structure of the input and output in GAMS right.

(Refer Slide Time: 13:19)



So, the various components in input are sets, data, variables, equation a model and solve statements and then we can have display statements. So, this display statement is optional right. So, for sets data which can be in terms of parameters tables or scalars and in variables right, we need to first declare it and then assign values and members to it.

Similarly, for equations we need to first declare what are the equations and then define the equation. We will use model and solve statements to build a model and then to subsequently solve it. Display statement can be used to display the results in a specified format by default GAMS displays a large number of output, but if we want to display a specific output we can use this display statements. For variables, we can also specify the bounds and the initial values.

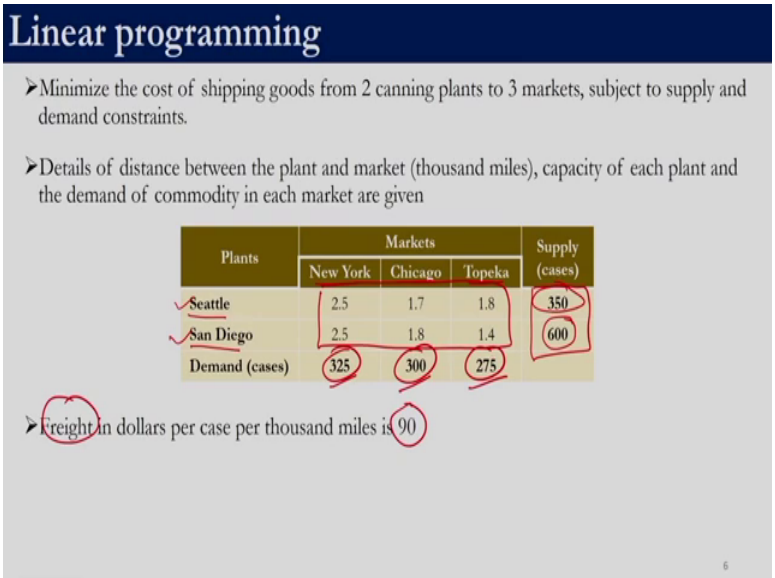
So, it is not mandatory, but if we have an initial starting point, then that can be provided to GAMS right; similarly if we have bounds that can also be given. So, these are the components of input, for output we will get something known as list file and a log file. So, a list file will have echo prints which is nothing but the reproduction of the model. We will have reference maps equation, listings status, report and solution reports. So, when we have a list file we will look into each of them.

There are two player lists which are available in GAMS channel; one is an introductory series which has three lectures and the other one contains advanced features of GAMS right.

(Refer Slide Time: 14:52)

The links are provided over here.

(Refer Slide Time: 14:55)

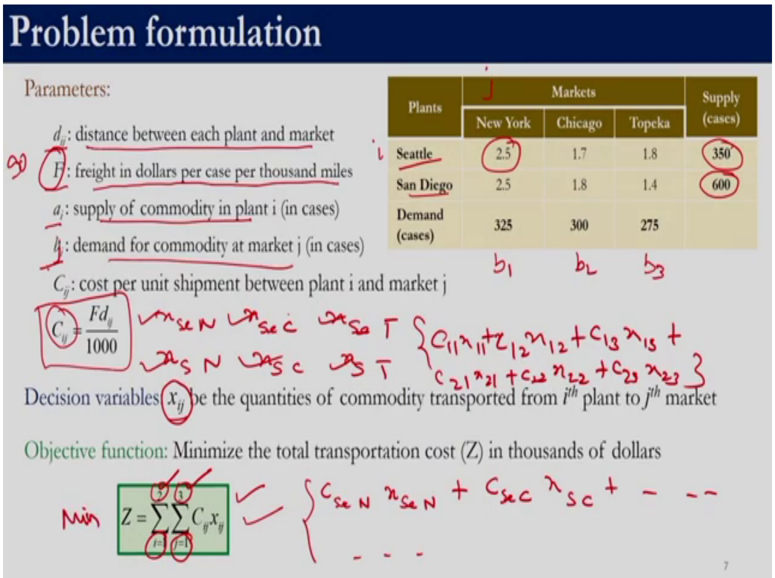


So, we will take this small example a linear programming example and we model this problem and we will see how to code it in a GAMS right.

So, these are 3 market places New York, Chicago and Topeka and there are 2 plants; one in Seattle one in San Diego right. So, the demand for a product in New York is 325 in Chicago it is 300 and in Topeka it is 275. The number of cases which the plant in Seattle can ship is 350 and the number of cases which the plant in San Diego can shift is 600 right. So, these values are the distances between the plant and the market in thousands of miles.

So, the optimization problem is that we need to satisfy the demand at these three places at New York, Chicago and Topeka; by shipping either from Seattle or San Diego. So, the optimization problem is to ship cases from the plants to the market, such that the demand is satisfied at all the three markets right. And the supply constraint is not violated and the total cost involved in transportation should be minimum. So, the freight causes given and the distance is known right. So, we need to satisfy the demands at these 3 markets without violating the supply constraint and minimize the cost that is the optimization problem over here.

(Refer Slide Time: 16:09)



Let us use the notation dij to denote the distance right, let us use the notation i for the plants and j for markets right. So, distance between a plant and market would be indicated by dij. So, d 1 1 can be the distance between Seattle and New York which is 2.5. F is the freight in dollars per case per thousand miles, which is given as 90; ai is the supply of commodity at plant i right.

So, a 1 is 350 and a 2 is 600 right; because that is the maximum number of cases which can be shipped from Seattle and San Diego respectively. So, this is bj right. So, bj is the demand for the commodity at market j right. So, b of 1 is 325, b of 2 is 300 and b of 3 is 275. So, the cost for shipment is given by this expression that the cost of shipping from i th plant to the j th market is f into dij by thousand right.

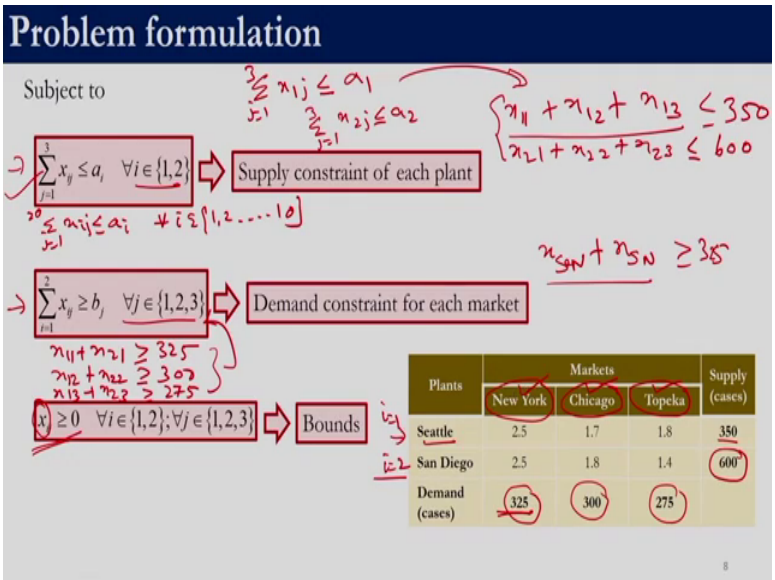
So, F is given us 90 in this case dij is given over here the distances are known. So, divided by 1000 will give us cij right. So, in this case, we will have to first select the decision variables right. So, decision variable can be xij. So, xij indicates how much quantity is supplied from one plant to one particular market right. So, the decision variables are how much is transported from Seattle to New York, how much is transported from Seattle to Chicago, how much is transported from Seattle to Topeka. And how much is transported from San Diego to New York, how much is to be transported from San Diego to Chicago, how much is transferred from San Diego to Topeka right, these are the 6 decision variables that we have right.

So, that we indicate by xij i is for the plants and j is for the markets. So, this expression gives us the total cost. So, what we are saying is, the cost of transporting from Seattle to New York into the quantity that is transferred from Seattle to New York, plus the cost for transporting from Seattle to Chicago into the amount that is transported from Seattle to Chicago and similarly the other terms right. So, we will have 6 terms.

So, instead of writing this expression we can compactly write it as sigma i equal to 1 to 2 right. So, this is for the plants and then this is for the cities right for j is equal to 1, 2, 3 right. So, if we basically expand this it will be c 11 x 11 c 12 plus, c 12 x 12 plus, c 13 x 13 plus, c 21 x 21 plus, c 22 x 22 plus, c 23 into x 23 right.

So, instead of writing this 6 terms, we can compactly write it like this. So, no matter how many cities and markets are involved, this expression would remain the same; we just need to change instead of this 2 and 3 with the respective number of markets and plants. So, now, we have this objective function. So, this has to be minimized.

(Refer Slide Time: 16:22)



So, now let us look at the constraints right. So, here we have two types of constraint that whatever is shipped from Seattle right, it should be less than or equal to 350; remember the constraint is not that we need to ship 350 from Seattle. The constraint is that we can ship a maximum of 350 cases from Seattle right. So, what is getting shipped from Seattle to New York is, x 11 ; what is getting shipped from Seattle to Chicago is, x 12 and what is getting shipped from Seattle to Topeka is x 13 right. So, this is what is required from Seattle. So, this has to be less than or equal to 350.

So, that is what is compactly written over here summation over j right for each plants. So, that is what is written for each plant right. So, the plant in Seattle so, this is i is equal to 1 and this is i is equal to 2. So, for each plant we are writing one constraint right. So, the first constraint is summation j is equal to 1 to 3 x 1 j is less than equal to a 1. So, a 1 in this case is 350. So, basically what we get is what we have over here right.

Similarly, the second constraint would be summation j is equal to 1 to 3 x 2 of j is less than or equal to a 2. So, the second constraint would be x 21 plus x 22 plus x 23 would be less than or equal to a 2 right a 2 in this case is 600 right.

So, these are the two supply constraints right. So, we can compactly write it using this expression. Again let us say if there are 10 plants and let us say 20 markets. So, this will just become j is equal to 1 to 20 xij is less than equal to ai for i belongs to 1, 2 all the way up to 10.

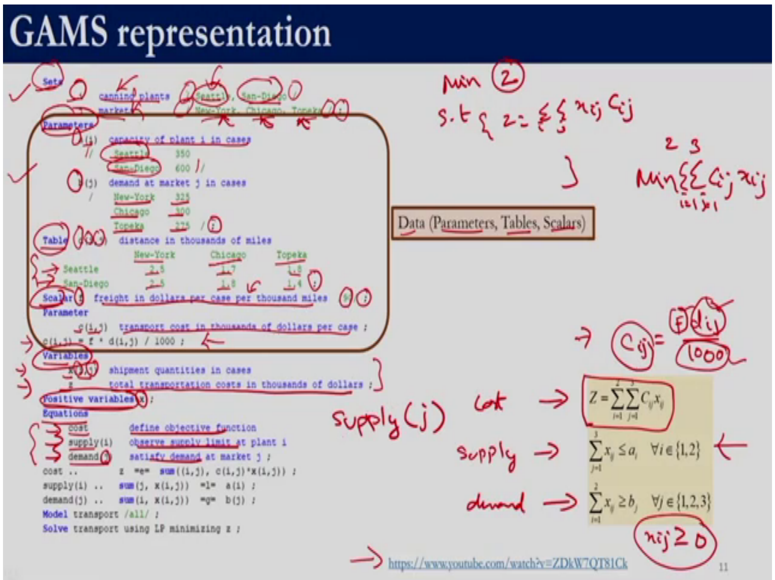
So, now that we have taken care about the supply constraint. We also have another constraint which says that the demand in New York is 325, the demand in Chicago is 300 and the demand in Topeka is 275 right. So, again we need to remember that it is not exactly 325 cases are required it is at least 325 cases are required.

So, we can supply more than 325, but not less than 325. So, in this case for New York we need to write a constraint, for Chicago we need to write a constraint for Topeka we need to write a constraint. So, that is why we have for all j belongs to 1, 2 and 3 right. So, we will write a constraint for New York for Chicago and for Topeka right. So, the amount that is coming from Seattle to New York, plus the amount that is coming from San Diego to New York so, this is the total amount of quantity which is coming to New York this should be greater than or equal to 325 right.

So, that is what has been written by this constraint for New York it is x 11 plus x 21 right summation over i should be greater than or equal to 325. Similarly we have x 12 plus x 22 right. So, this is the balance over Chicago. So, this has to be greater than 300. Similarly we have x 13 plus x 23 should be greater than or equal to 275 right.

So, these three constraints can be compactly written using this one expression right. And since xij is the number of cases which are transported from a plant to a market, it has to be greater than or equal to 0 right. So, its not necessary that every plant ships to a particular city. So, as long as these constraints are respected any number of cases can go from any particular plant to any particular market. So, now, we have this model over here that model is written over here.

(Refer Slide Time: 23:08)



So, this is the model we have in addition to this we have xij which is greater than or equal to 0.

So, the discussion that what we are going to have here is also available here as part of GAMS help right. So, you can also look at that discussion. So, first we will identify the number of sets which are present in this problem. So, in this case, there are two sets which are available right. So, we use this keyword sets to define two sets, the name of the first set is i the name of the second set is j right. So, i and j are the two sets which you are using. So, i we are going to use it for denoting the plants and j we are going to denote it for markets.

So, the name of the set is i and j. So, we need to give two values for i because we have two plants one in Seattle and another one is San Diego similarly we have 3 markets New York Chicago and Topeka right. So, these are the values over here. So, once we are done with defining sets, we need to provide this semicolon right and the values are to be given within these slash operators. So, starting and ending right. So, New York if you remember we had space in New York right. So, spaces are not allowed over here. So, we have used New hyphen York.

So, we have first defined sets right, we have declared what are the sets. So, we have two sets i and j and then we have assigned the values for those sets right. Now we will provide the data whatever we have right. In GAMS data can be in terms of parameters tables or scalars right. So, parameters is usually used for vectors right, table is for usually providing data in the form of a matrix and scalars are constant single values right. If you recollect the problem you would know that we had constraints on Seattle and San Diego as to how many cases it can transport right.

So, here what we are doing is, we are defining a parameter a it runs over the set i right. So, set i means a of Seattle and a of San Diego right. So, what we have over here Seattle and San Diego should come over here Seattle and San Diego right. This canning plants this market is more like a description, we can give any values and it can also have spaces right. So, for example, here we are seeing a is capacity of plant i in cases right and again within this slash operator we provide the values right; for Seattle it is 350. So, we write Seattle because Seattle and San Diego are the elements of i.

So, we are seeing a of Seattle is 350 a of San Diego is 600. Similarly we use another parameter b right this parameter is for every element in the set j right. So, in this case j is nothing, but the markets. So, we have 3 markets New York, Chicago and Topeka right and these are the elements of this set j New York Chicago and Topeka.

So, we write New York, Chicago Topeka and the appropriate values which we have 325 300 and 275. So, once all the parameters which are required are defined we need to end it with a semicolon over here. So, in MATLAB you would put semicolon every line and even if you do not put semicolon you would not get an error it will just print things in the command window also; over here you need to compulsorily put a semicolon right.

So, we have defined two sets, we have given the parameters with respect to the markets and the plants right. Now we need to provide the distance. So, the distance if you see it is in the form of a table right. So, we use this cable or table. So, sets parameters and tables are keywords. So, the name of the variable is d right and d is i comma j. So, the rows indicate the plants and columns j indicate market right. So, we have two plants which are the rows. So, here we have in rows Seattle and San Diego and we have three columns New York, Chicago and Topeka.

So, here it is New York, Chicago Topeka; these are the elements of the set. So, GAMS understands that i when we refer to i we are referring to this i th set i and set i contains two values Seattle and San Diego. So, it expects two rows over here similarly for markets we refer it using j right. So, and there are 3 markets. So, three columns the columns should be New York, Chicago and Topeka right. So, these are the values which are given to us. So, we directly write those values and once we are done defining table we need to again end it with a semicolon right.

So, we have seen sets parameters and table. So, we can also provide data in terms of scalar right. So, Scalar is a keyword right over here we are defining a Scalar f. So, the description of f is freight in dollars per case per thousand miles right. So, this is a description and the values again entered between these two slashes. So, the value in this case is 90. So, once we are done defining scalars we again need to end it with a semicolon right.

So, now we have to calculate the cij right remember the expression cij cij is equal to F into dij by 1000 right. So, we have defined this dij we have defined F and this is 1000 is a constant value. So, now, we need to calculate this cij remember this is not a constraint right; because this does not have any decision variable dij is known F is known 1000 is known cij can be uniquely calculated right. So, that is what we are going to do now.

We are defining a parameter c it runs over the set i comma j, this is the explanation for cij and here we write the expression for calculating cij. So, c of i comma j is equal to f which is a scalar which has been defined into d of i comma j which is also defined divided by 1000 and we put a semicolon right.

So, whatever preliminary calculations are required, can be written over here itself. So, now, we have defined all the data which is required for this problem right. So, now, we will define what are the variables that are required here we have two types of variables. So, variables again is a keyword. So, we have variable x and z right. So, Z we will indicate to denote the objective function right. So, objective function is also considered as an equation in GAMS right. So, if it is an equation we need to have a left hand side to this.

So, instead of minimize summation ci j x i j i is equal to 1 to 2 and j is equal to 1 to 3. What we are doing is we are saying minimize z right subject to whatever the rest of the constraints are right and in addition we also have this equation which says z is equal to summation over i and summation over j xij cij.

So, that is why we have this variable right. So, this variable z will indicate the total transportation cost in thousands of dollars right. So, that is the purpose of this variable z this xij is the decision variable which we discussed earlier right. So, x of i comma j. So, once we write i comma j GAMS understands that i which we have defined is a set. So, it has the values of Seattle and San Diego and j which we have previously defined as a market has the values of New York Chicago and Topeka.

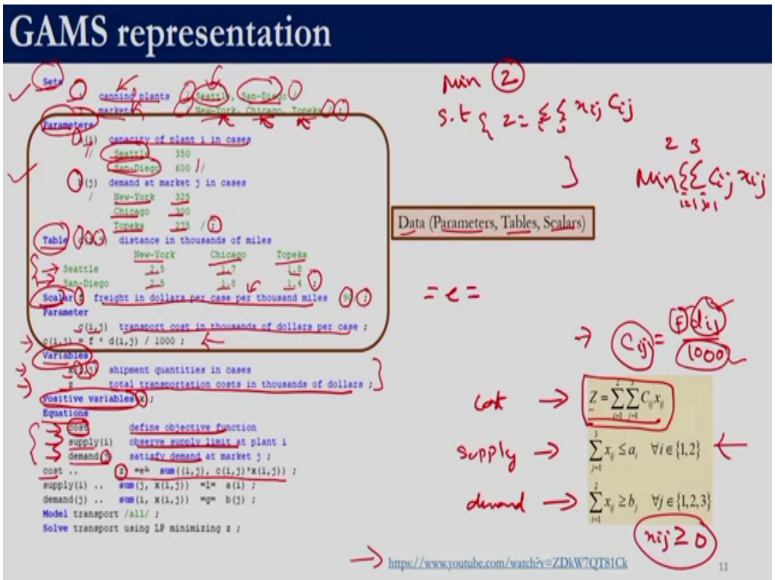
So, this is how we define the variables; since we have this requirement we write positive variables x right. So, positive variable this is again keyword right. So, it indicates to GAMS that x can take only values which are greater than or equal to 0. So, we do not have an upper bound on x right. So, that is why we do not need to give over here and then we have three equations over here. So, this is the equation for supply this is the equation for demand right.

So, let us call this as equation for the total cost right. We have three equations we need to first define the name of the equations and then define the equations right. So, here we use the key word equations and write cost supply of i demand of j and write whatever description that we want for each of them right.

So, now GAMS will be able to interpret that cost is just one equation because there is no indexing supply is for every element in set i. So, set i contains two values Seattle and San Diego. So, when we write supply of i GAMS understands that supply of Seattle supply of San Diego; because we have two values similarly demand of j right. So, j is a set we have earlier defined. So, this will be demand of New York demand of Chicago demand of Topeka right. So, these three are the name of the equations and how many equations are there can are also specified using indexing right.

So, if we write supply of j instead of supply of i right, let us say we write supply of j in this case it will have three constraints supply will have three constraint supply at New York supply at Chicago and supply at Topeka.

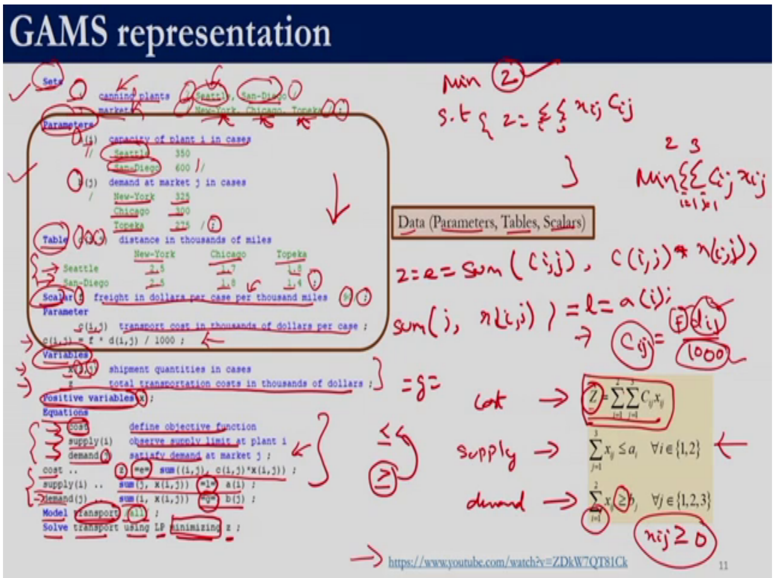
(Refer Slide Time: 32:21)



Now, we need to define the equations right. So, here we define the equation. So, we have used the equation name cost. So, we are writing what is the equation cost.

So, if we write cost over here and then space two dots right and then we write whatever constraint we have over here. So, here we have this equation to be written. So, the equation is the variable z this equal to e equal to means it is an equality constraint right. So, that is why we have equal to e equal to over here. So, to write this expression right. So, we are going to sum something.

(Refer Slide Time: 32:54)



So, let us say z is equal to e is equal to right. So, we write sum right and we are going to sum something right.

So, we are going to sum with i as well as with j right. So, sum of i comma j. So, sum of i comma j then this is the index on which we want to sum then we need to have a comma and then the expression right. So, c of i comma j star x of i comma j that is what is written over here right. And we need to end the statement again with semicolon then we write the equation supply of i right. So, supply of i here we have a less than or equal to constraint right and we have summation over here.

So, what we have is, summation we are going to sum something and it is going to be over j right. So, sum j comma the expression. So, the expression in this case happens only to be x of i comma j right. So, this is expected to be less than or equal to a of i right. So, less than or equal to constrain in GAMS is written as equal to l equal to a of i we ended with a semicolon right. So, this is the equation for supply similarly demand of j we are going to sum it over i right.

Because here we have summation over i comma x of i comma j right is equal to g because we have greater than or equal to over here right. So, equal to g equal to indicates that the left hand side has to be greater than or equal to the right hand side is equal to g equal to b of j right. So, we have defined all the three constraints right. So, the good thing over here is unlike MATLAB where we were forced to give all the constraint as less than or equal to form right. Here we can give less than or equal to constraint as well as greater than or equal to constraint directly.

We do not need to transform these constraints into these type right, which was the case in MATLAB; If it is an equality constraint we directly use equal to e equal to if it is less than or equal to we use equal to l equal to if it is a greater than or equal to constraint we will use equal to g equal to right. So, now, we have defined all the constraints right; now we are building a model right we use the keyword model the name of the model. So, here we choose to give the name transport right.

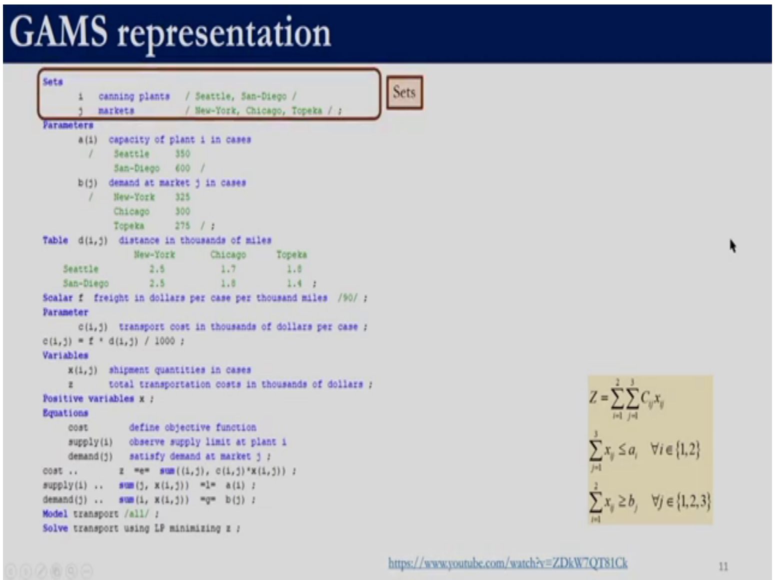
And within this clash we give all. So, all is again a keyword so; that means, all the constraints which have been defined in the current GAMS file right, are part of the model transport right. So, for example, if we do not want to have demand constraint one way is to remove this or comment it right. The other way is to build a model transport one letter without including the demand constraint we will see that a little bit later. So, for the current case we are building a model right the name of the model is transport.

And we are including all the constraints which have been defined in the current GAMS file right. So, building model does not solve it we need to explicitly solve using this keyword solve right. So, we want to solve the name of the model using has to be provided right and then the type of the problem in this case if we see this problem is a linear programming problem. So, we provide LP and we want to minimize the objective right.

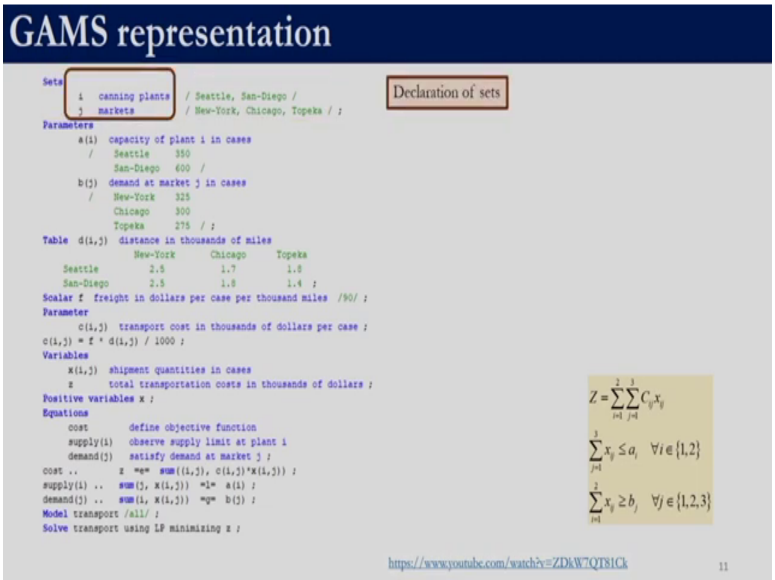
So, we explicitly say minimizing and then we say the variable z because that is what we want to minimize right. So, if we have a maximization problem then instead of this minimizing over here we just need to give maximizing right. So, again unlike MATLAB where in you are force to solve only minimization problem here you can solve the maximization problem as well as minimization problem without necessarily transforming the maximization problem into a minimization problem.

So, this completes the description of the GAMS file. So, this has to be written as a GAMS file. And if we execute this particular file we will be able to solve the transport problem. So, now, let us quickly go through the file once again right.

(Refer Slide Time: 36:57)

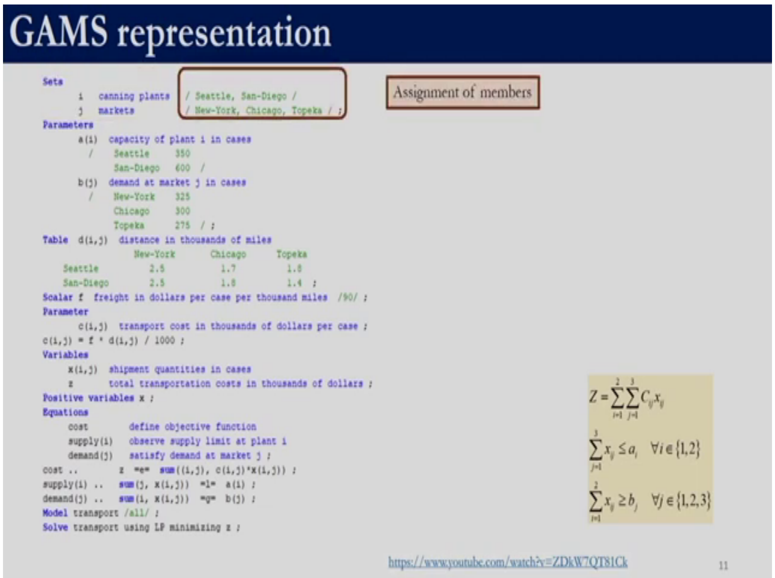


(Refer Slide Time: 37:00)



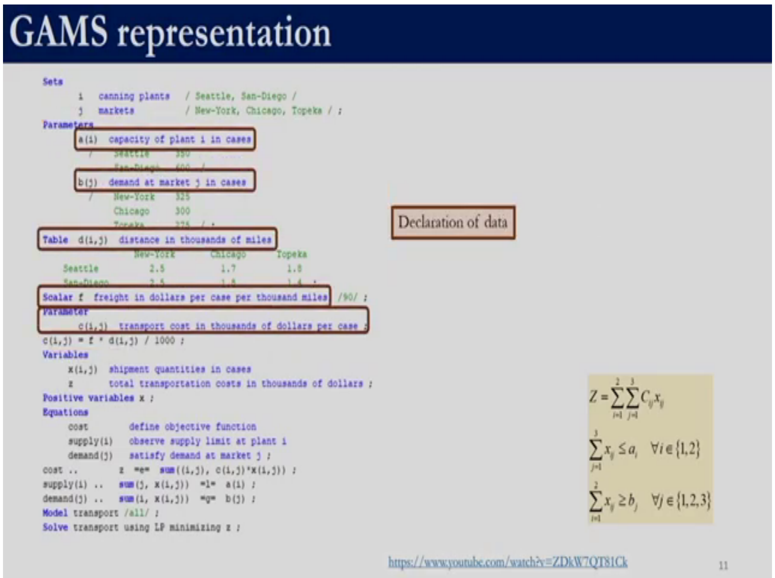
So, initially we define what are the sets which are there.

(Refer Slide Time: 37:01)



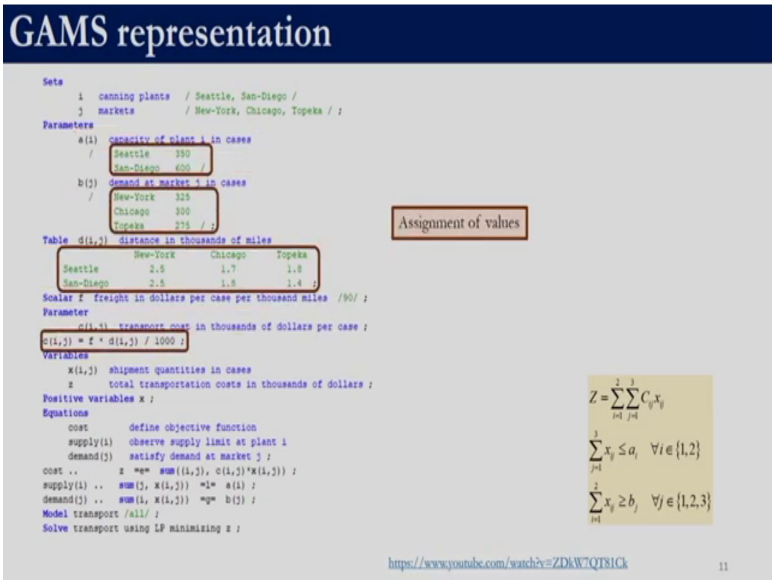
And then assign the values; then we have this data in this case data is given as parameters tables as well as scalar.

(Refer Slide Time: 37:08)



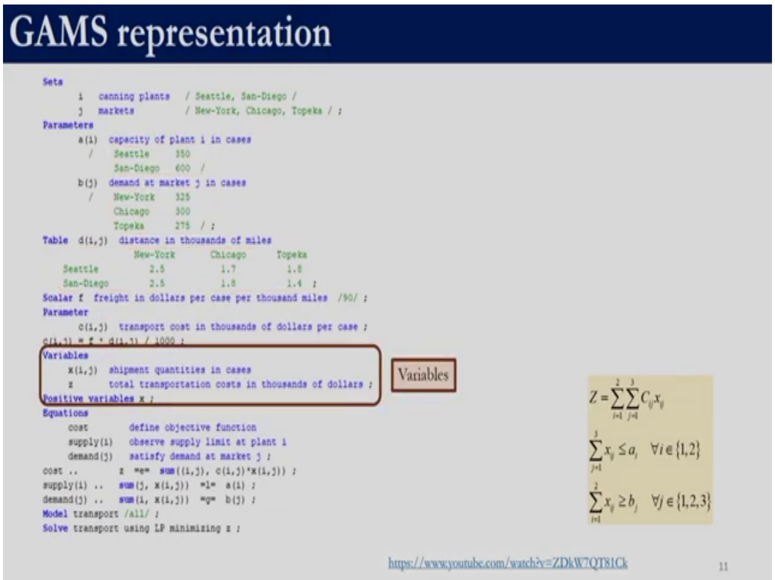
So, over here it is declaration of the data we are just declaring that there is a variable a b d f and c right.

(Refer Slide Time: 37:15)

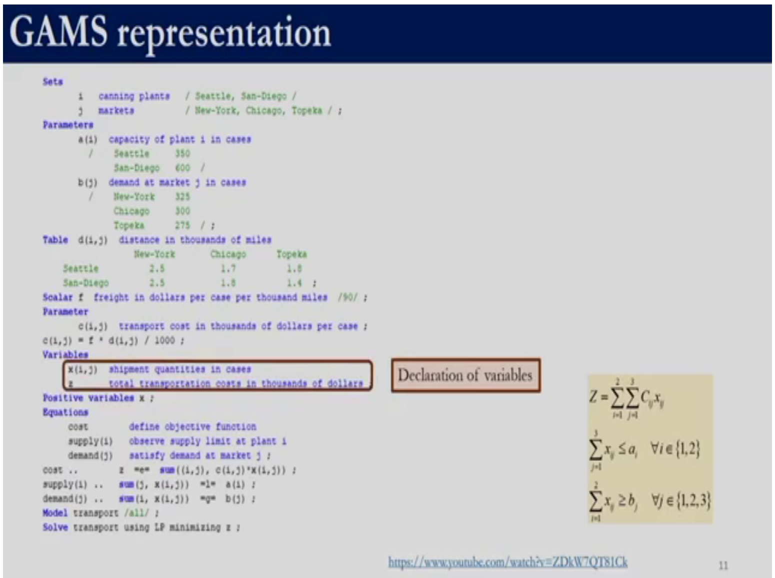


And then here we are providing the values for the parameters.

(Refer Slide Time: 37:19)

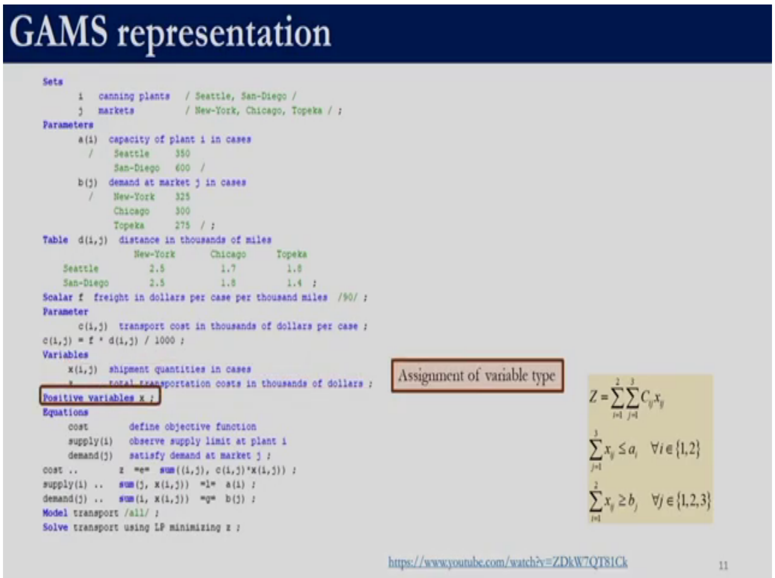


(Refer Slide Time: 37:22)



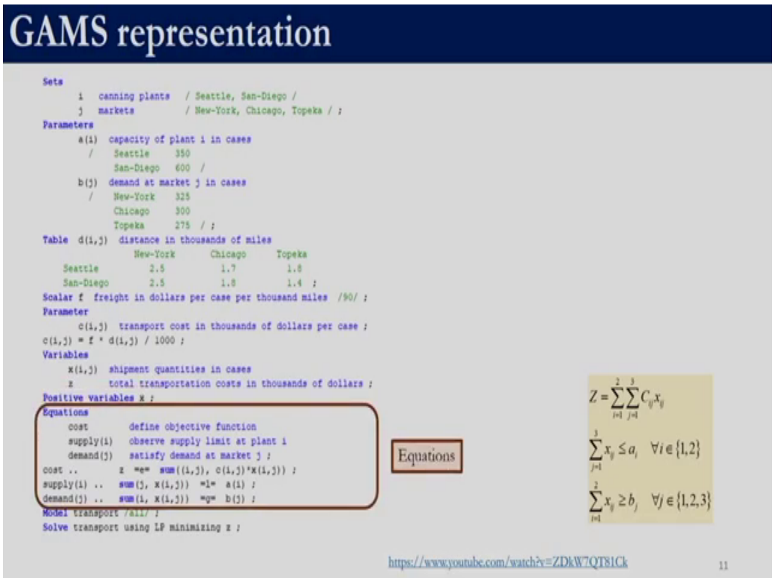
And then over here we are defining the variables we are declaring the variable x and z along with its description.

(Refer Slide Time: 37:26)



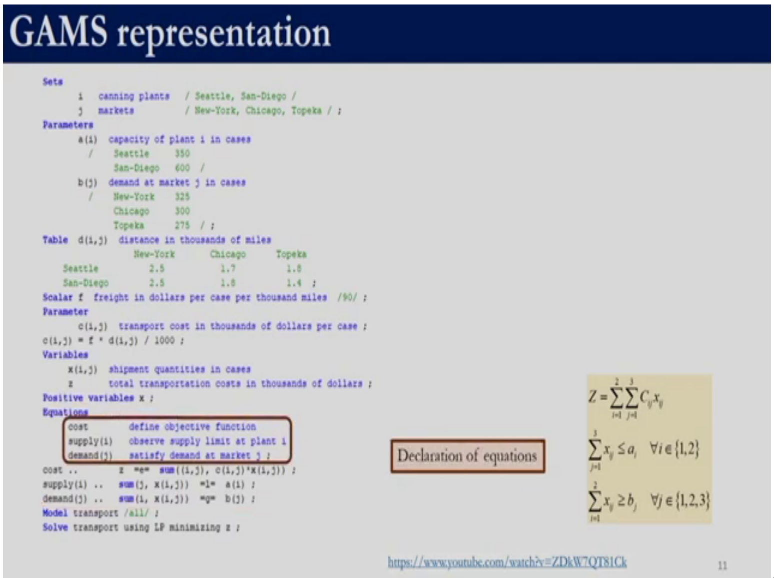
Here we are assigning the type of variable right. So, Positive space variables x ensures that x will take only values which are greater than or equal to 0.

(Refer Slide Time: 37:35)



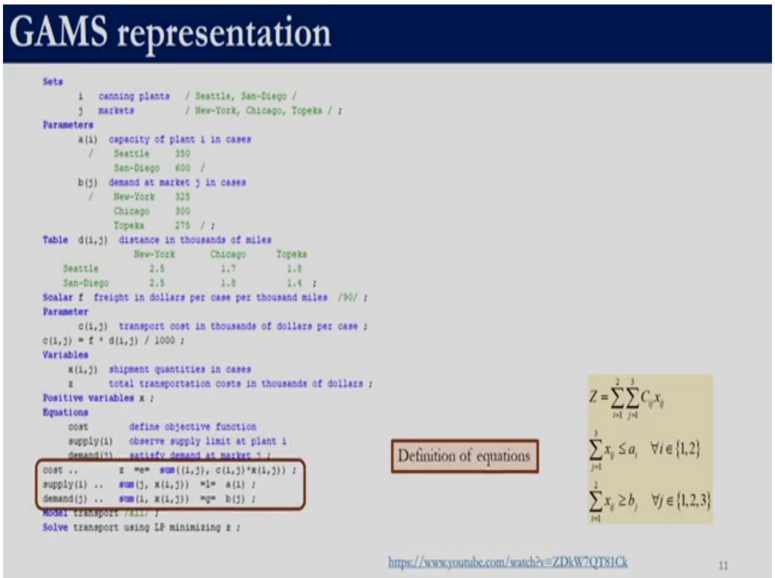
Then we have the equations block. So, we first write the keyword equations and then list the set of equations.

(Refer Slide Time: 37:45)

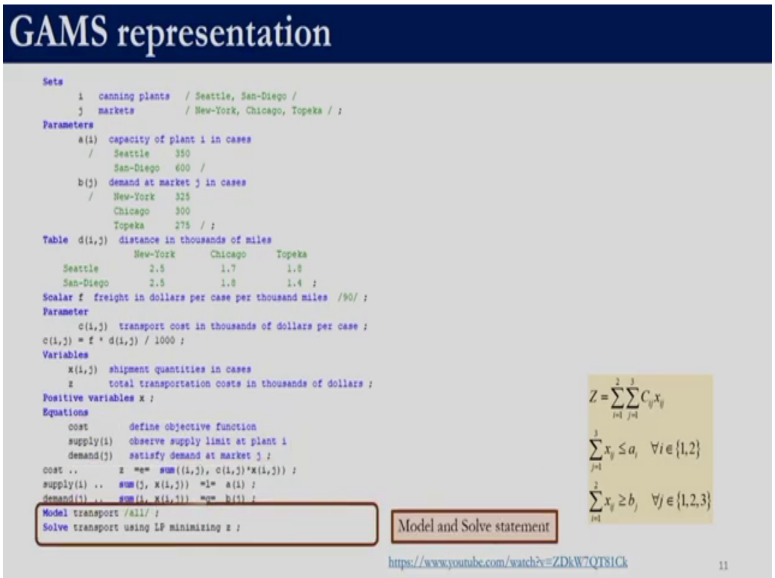


Then we have the equations block, we declare the equations and their appropriate description.

(Refer Slide Time: 37:50)

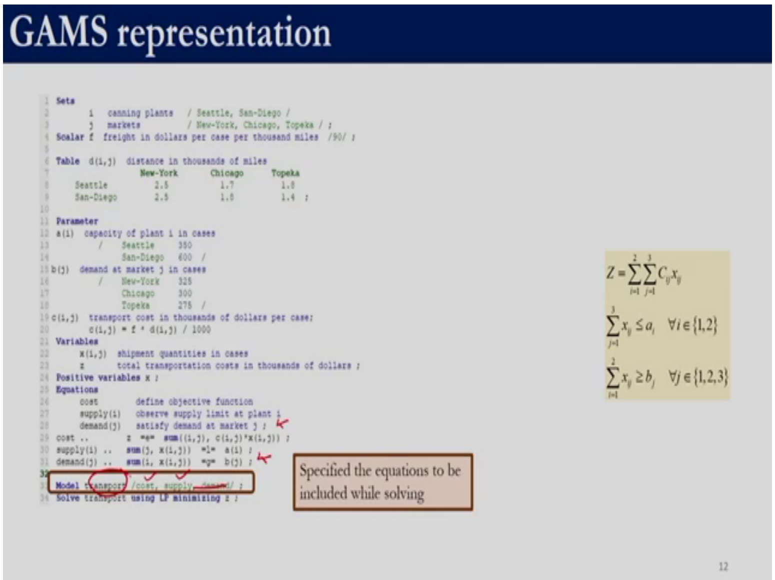


(Refer Slide Time: 37:51)



And then we define the equations this is followed by building the model right here the name of the Model is transport. And it includes all the constraints listed in this GAM file; Model is a keyword. In the next line we have Solve which is a keyword. So, we are solving the Model transport using linear programming with minimization of z.

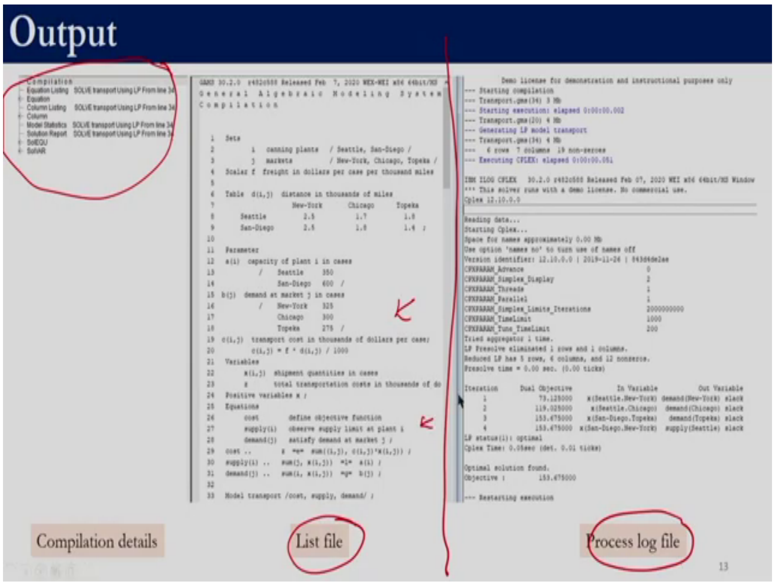
(Refer Slide Time: 38:12)



So, the statement all which we had previously basically means cost comma supply comma demand; because those are the three constraints. So, for example, if we do not have demand constraint, then we can remove this comma and demand right. So, in this case though the equation demand has been defined and assigned right, it will not the model transport will not contain the demand equation; it will only contain the equation related to cost and supply.

So, here we can list out hundred equations and depending upon what we are solving we can include a selected set of equations and can build multiple models in the same file.

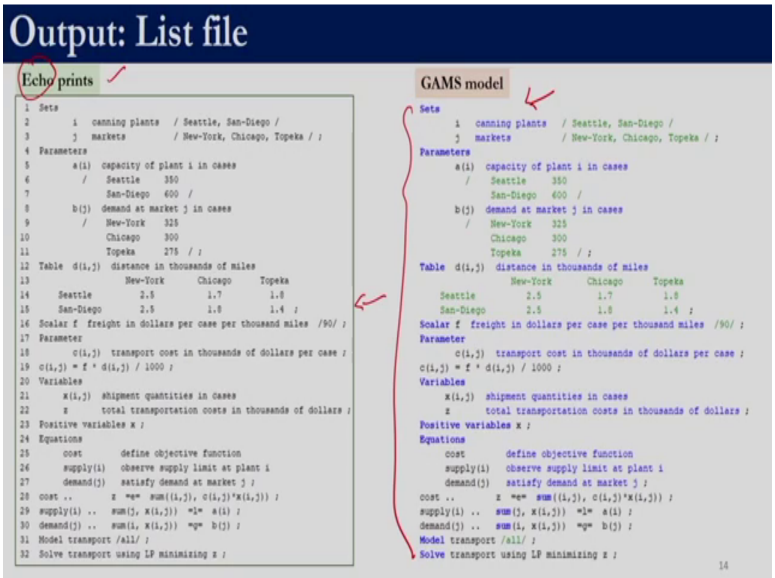
(Refer Slide Time: 38:49)



So, when we execute that GAMS file right this is the output that we would get right. So, basically we will get two types of file; one is called as List file and another one is called as Process log file right.

So, list file will contain two parts. So, on the left hand side we get the contents of the list file, by clicking any of this we can go to the appropriate section in the list file. So, the list file is usually a very long file. So, we can jump to the appropriate section, in this section we can click on whatever is required. So, for example, if we want model statistics or solution report, we can directly click on that and we will be able to see that particular section in that list file.

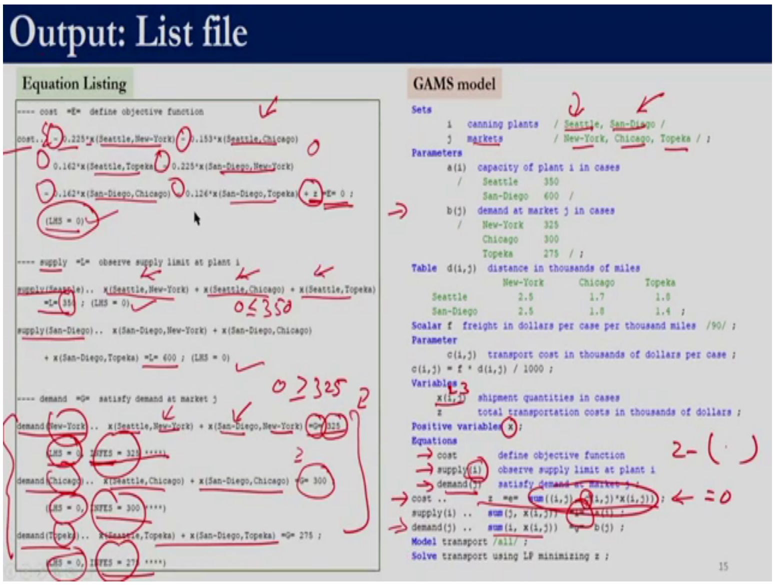
(Refer Slide Time: 39:25)



So, now let us look into the list file. So, the first section of list file has these echo prints right. So, this is an example of list file for the model which we have built right. So, this is our GAMS model, when we execute we get a list file the first section is the Echo prints. So, over Echo prints if you see, it is a exact replication of the model that we have, that is why it is called as Echo right. So, if you want you can switch it off, but by default it is on right. So, once we save the list file at any point of time, we will also be able to know what is the model which we solved for the results in the list file. So, this is a big advantage of list file.

Unlike many other software in which results are usually stored away from the program which generated it in GAMS; as part of list file we will get both the model as well as the results in the list file.

(Refer Slide Time: 40:13)



So, list file also has this Equation Listing right. So, this is our model this is the equation a listing right we have these three equation cost supply and demand. So, over here the equations are listed that is why it is called as Equation Listing.

So, what we wrote over here cost is z is equal to e is equal to sum of i comma j comma cij into x ij what we have written over here. So, the expanded version of this is given over here. So, cij if you calculate it would come to 0.225, then the variable x which indicates Seattle to New York Seattle to Chicago Seattle to Topeka, San Diego New York San Diego Chicago and San Diego Topeka right.

So, that is what is over here. So, what is displayed is? z minus this part equal to 0 right. So, that is why we have z over here which is positive and the rest of the terms have a negative sign right and the right hand side is equal to 0. So, this is the cost equation let us look at the supply equation right. So, we have supplies over set i right and i contains two values Seattle and San Diego. So, here we have supply of Seattle and supply of San Diego this constraint is actually less than or equal to constraint right. So, that is why we have less than or equal to 350 and less than or equal to 600 right.

And the variable if you see it is sum over j right and j is our market right. So, supply of Seattle is nothing but x that is going from Seattle to New York; x that is going from Seattle to Chicago and x that is going from Seattle to Topeka. So, that has to be less than or equal to 350. So, similar to supply Seattle, we also have supply San Diego right. Then the third set of equation is demand; demand runs over j right.

And j we have three values in demand. So, demand of New York demand of Chicago and demand of Topeka. So, over here this equation is sum over i X of i comma j right. So, since this this is the j this remains constant. So, whatever that is coming from Seattle and whatever that is coming from San Diego to New York should be greater than or equal to 325. Similarly for Chicago whatever is coming from Seattle and whatever is coming from San Diego has to be greater than or equal to 300 and the third constraint right.

So, we do not need to explicitly specify these equations when we are writing model. Model we wrote in a very compact way right similar to what we would develop using a pen and paper that is the advantage of GAMS; that the model which we develop in algebraic form can almost be directly coded into GAMS.

So, since in this problem we did not give default values. So, the default values of all these 6 variables. So, there are 6 variables because x is defined across i comma j and i is two and j is 3. So, basically we have 6 variables. So, since all the 6 variables are positive and we have not given any initial guess right. So, by default takes value of 0 for all the decision variable if we do not supply a initial value right. So, if a value of 0 is taken the left hand side will come out to be 0.

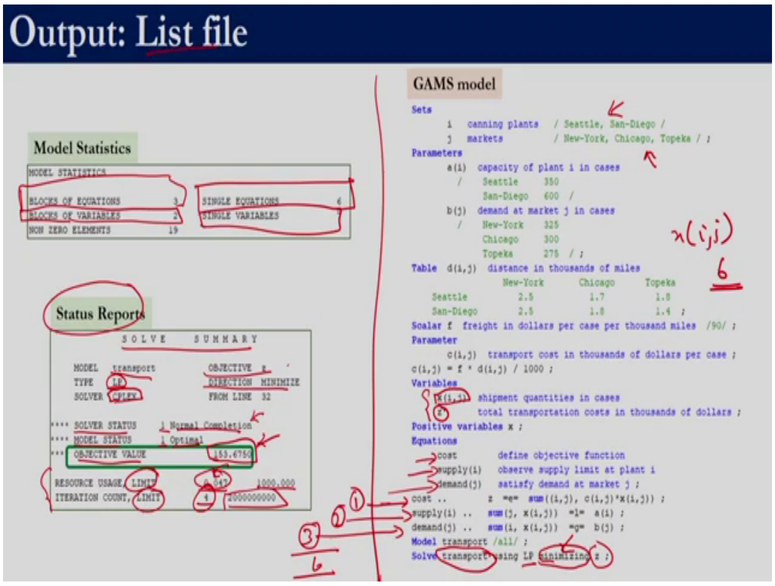
So, that is why we have LHS is equal to 0 and since 0 is equal to 0 this constraint is satisfied. So, it does not indicate that there are any infeasibilities right. Similarly these two equations also if you see the value 0 over here right then all of this would be 0 the left hand side is 0 and the right hand side is 350. So, 0 is less than equal to 350 is what we currently have because the initial values are taken to be 0. So, this constraint is satisfied similarly this constraint would also be satisfied.

However if you look at into these three constraint right. So, here if we take 0 0 the left hand side is 0, but the right hand side is 325 and this is a greater than or equal to constraint right. So, 0 is greater than or equal to 325 that is the current state; since we have not supplied any initial value GAMS takes it to be 0. So, in this case we have lhs to be 0 greater than or equal to constraint and 325; this is what indicates that we have infeasibility over here.

So, in all the three cases the left hand side if you see it will come out to be 0 and the right hand side is a positive number and the constraints are of greater than or equal to right. So, all these three constraint at the initial point which is 0 0 0 is an infeasible point right. So, this equation listing does not give you the result this equation listing tells us what is in the initial phase.

So, we have seen two components of list file; the first one was Echo print right which is just reproducing the model and then the Equation listing right.

(Refer Slide Time: 45:07)



The other part of list file is model statistics. So, model statistics tells us about the equations and variable. So, here if we see we have two types of variables right. So, that is what is blocks of variable. So, we have two types of variable. So, one is indicated by x and the other one is indicated by z. And we have three blocks of equations right one is cost another one is supply of i and the other one is demand of j.

But the number of decision variable is x if you see it is just not one single variable, but it is i comma j right x of i comma j. So, i in this current case we have two values and j we have three values. So, basically x of i comma j has 6 values right; because there are two elements in i and three elements in j right. So, 6 variables plus another variable this is a single variable z. So, we basically have 7 variables. So, that is what is denoted here. And then over here if you see constraint this is just one single constraint supply of i is for Seattle as well as San Diego.

So, here we have two constraints and demand we have for three cities right. So, this will be basically 6 equations 3 plus 2 plus 1. So, that is what is given over here 6 equations; remember this side is the GAMS model right. So, we are just showing you the model. So, that it is easier to explain, but list file has only echo print the model is separate right. So, in status report we get the solution summary. So, the name of the model is transport because we solve the transport model right TYPE LP.

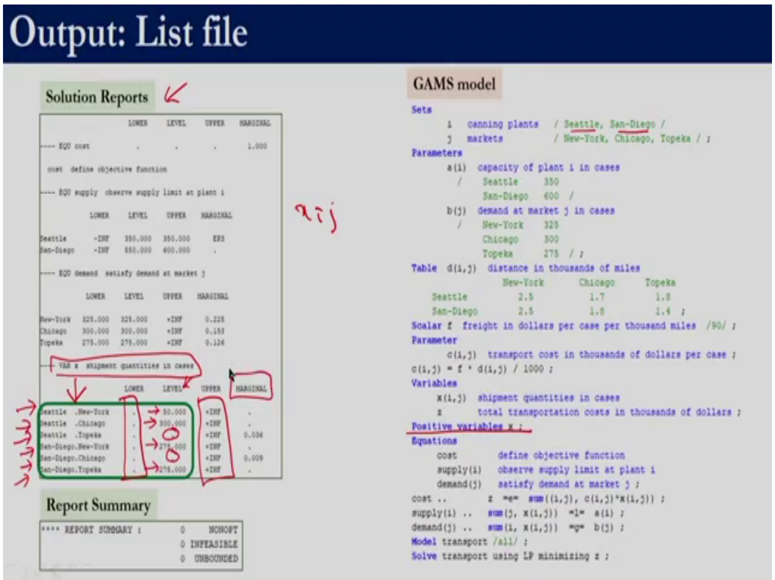
Because we solved it as linear programming problem the solver by default it took CPLEX we did not specify the solver. So, that is why it shows solver is CPLEX the OBJECTIVE is to minimize z and the DIRECTION is MINIMIZE. So, we since its a minimizing problem we got minimize over here had it been maximizing over here we would have got maximize over here right.

So, over here it says the SOLVER STATUS and the MODEL STATUS right. So, SOLVER STATUS 1 indicates Normal Completion and MODEL STATUS 1 indicates that it is the Optimal solution. And over here it gives the OBJECTIVE function VALUE. So, the value of z is 153.6750.

So, this section shows the amount of time that is used. So, in this case it is 0.047 seconds and it employed 4 iterations right. So, the default setting for time and the iteration count is 1000 and this value respectively. So, if you see that this value is also 1000 right so; that means, the amount of time utilized is also 1000 and the limit is also 1000. Then it might happen that it is not Normal Completion it completed; because of lack of additional resources same thing for iteration.

So, now we have seen 4 sections one is Echo print equation listing model, statistic status report right.

(Refer Slide Time: 47:59)



So, the decision variables are given in Solution Reports right. So, the value for variable x can be found here. So, over here it enumerates the variable right. So, for example, x i comma j right so, x i is Seattle. So, we will have x Seattle to New York x Seattle to Chicago x Seattle to Topeka and the second value is San Diego right. So, San Diego to New York, San Diego to Chicago and San Diego to Topeka, right.

So, these are the 6 decision variables which we have the value of the 7th decision variable we have seen right which was nothing but the objective function value for the 6 variable. The lower bound which we had as 0's right because we have given it as positive variable. So, the lower bound is 0, the upper bound is infinity again because it is a positive variable we did not explicitly specify any upper bound right. So, the lower bound is 0 since it is 0 GAMS does not write 0 it just puts a dot. So, Seattle to New York we need to ship 50 cases, Seattle to Chicago we need to ship 300 cases.

Nothing is being shipped from Seattle to Topeka because here it is a dot which basically indicates 0. So, from San Diego to New York 275 cases are to be shipped, from San Diego to Chicago nothing is being shipped and from San Diego to Topeka we need to shipped 275 right. So, this is how we get the values of the decision variables. So, being an introductory video over here we are not duelling into too much details, you can look into the help of GAMS and try to identify what are the other things that GAMS returns.

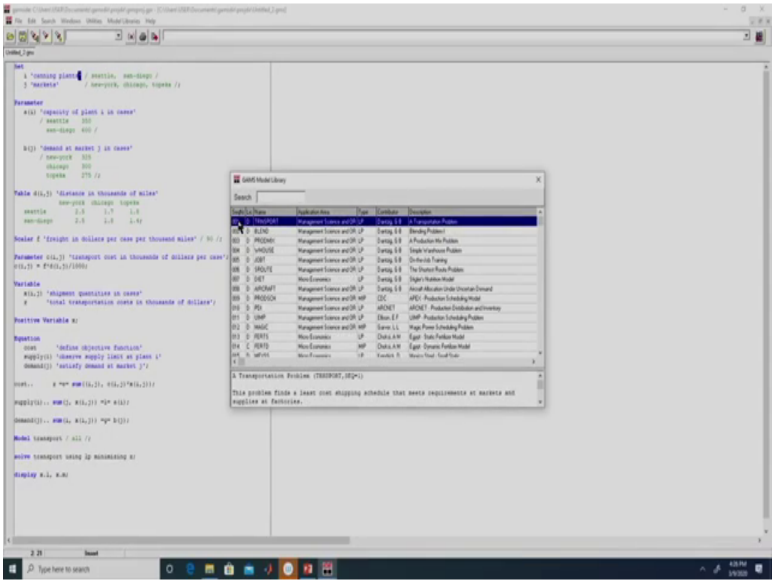
For example you can try to find out what is this marginal price and if it indicates the shadow prices. So, those are the major portions of a list file right.

(Refer Slide Time: 49:39)

We will now look into the Log file this is an example of the Log file. So, the Log file gives the location of the GAMS file, the version with which we are solving and the license details; with what settings it was executed. The Objective function value obtained in each iteration.

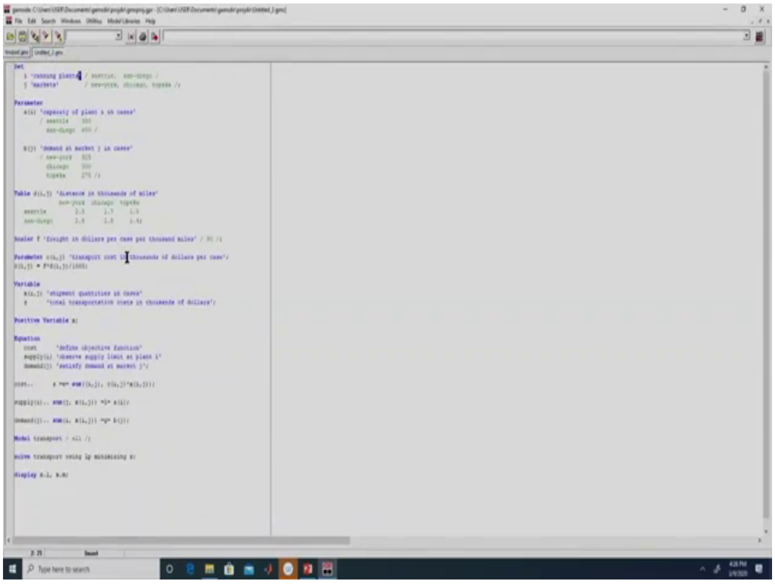
So, for example, here for the same problem we did get 153.675 in the 4th iteration right. So, that was given by the list file, but what is happening at every iteration was not given in the list file right. So, that can be obtained from the Log file; it also provides us with the time taken for completion the reason for termination of the algorithm and also the objective function value.

(Refer Slide Time: 50:22)



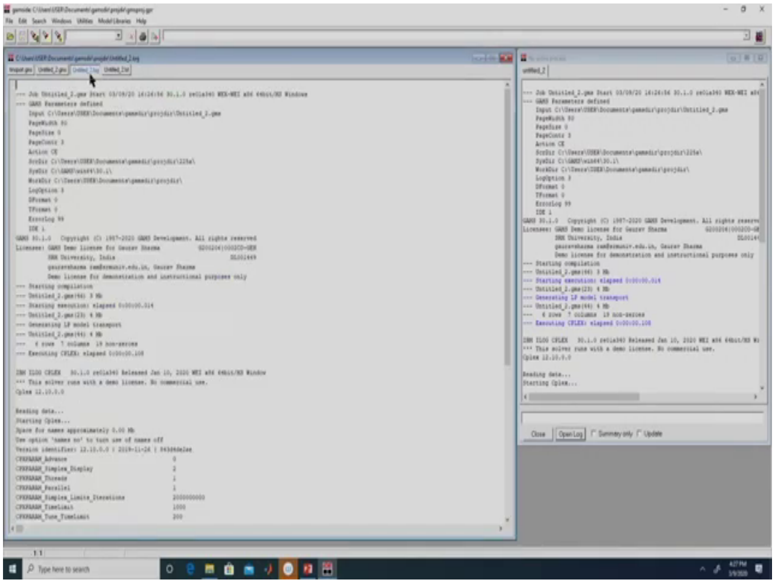
So, this is the GAMS IDE right. So, here if we go file new this is where we can write the GAMS code right. So, this is the GAMS code which we have discussed. So, this is also available over here in this grams model library, if you click on this sequence number the first example that you would get is the GAMS model right.

(Refer Slide Time: 50:32)



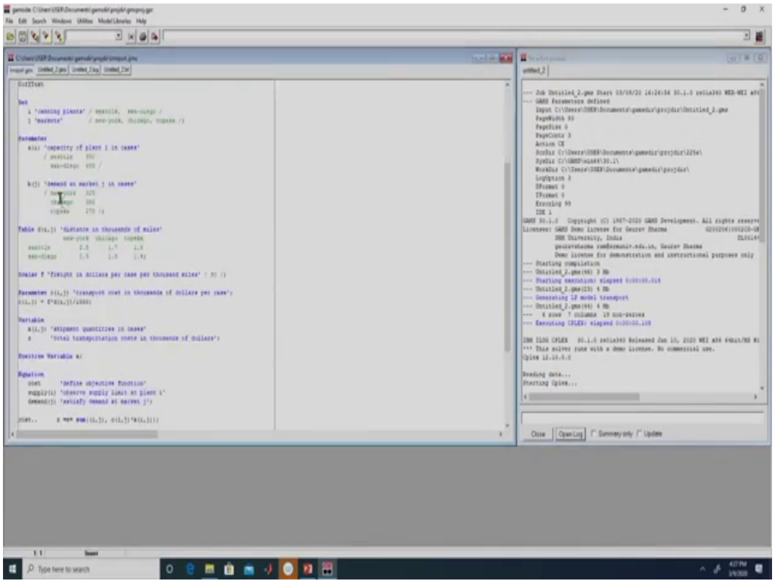
So, this is what we have copied over here right. So, if we now execute this you can either click on f 9 to execute it or you can click on this button right.

(Refer Slide Time: 50:44)



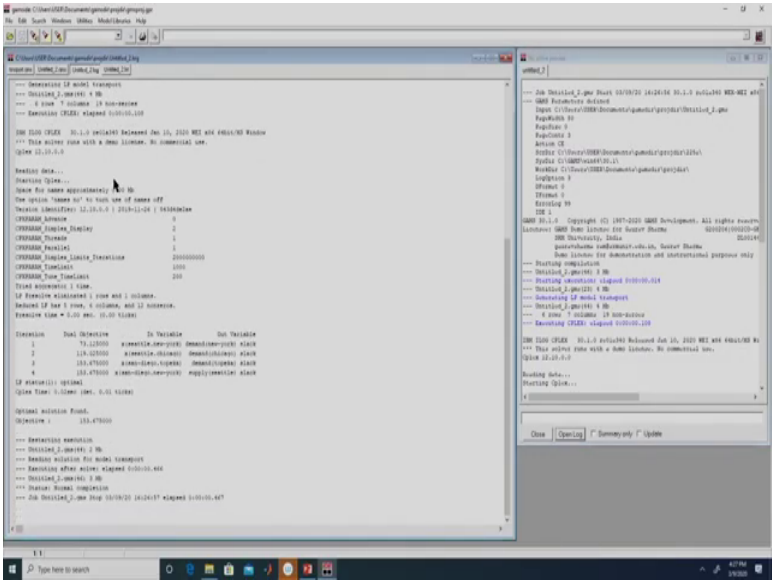
So, if you see here what we get is the log file right. So, if we click on this open log it will open log in the editor itself right.

(Refer Slide Time: 50:52)



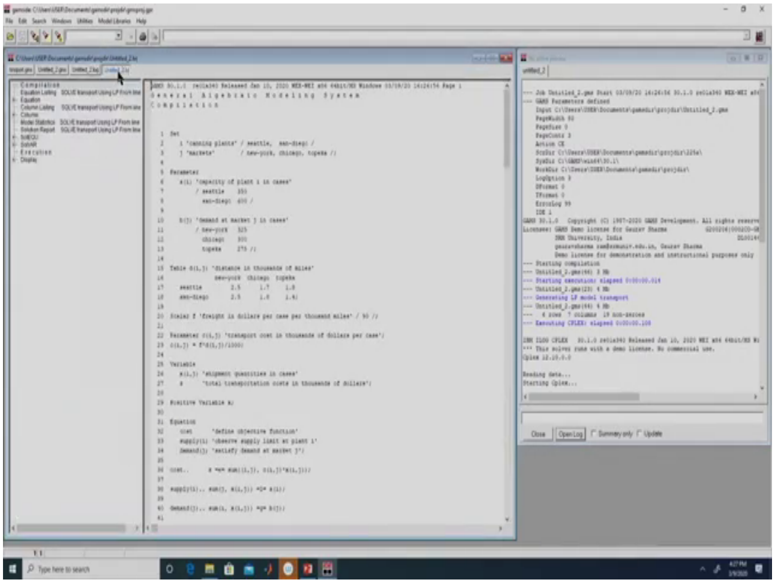
So, this is the editor right. And then we have this listing file untitled underscore 2 dot gms is the file that we executed the log file is untitled underscore 2 dot l o g right.

(Refer Slide Time: 51:04)



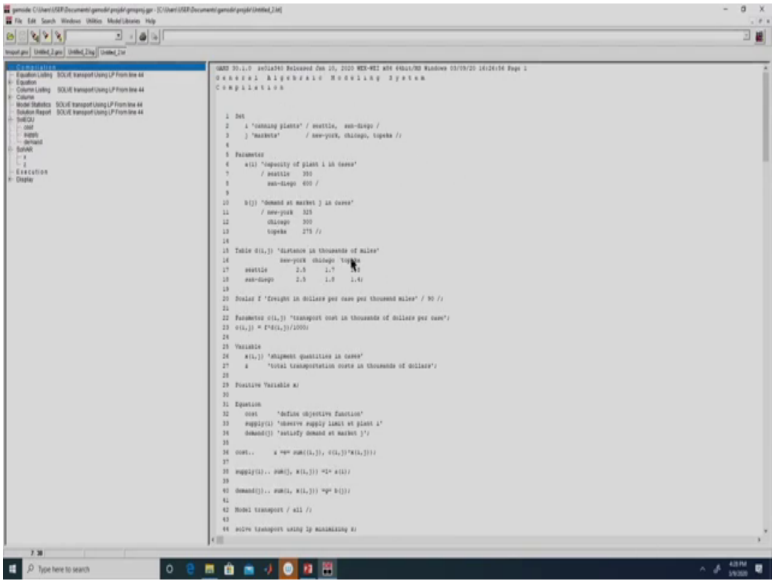
So, this basically gives the details which we have previously discussed and this is the listing file.

(Refer Slide Time: 51:08)



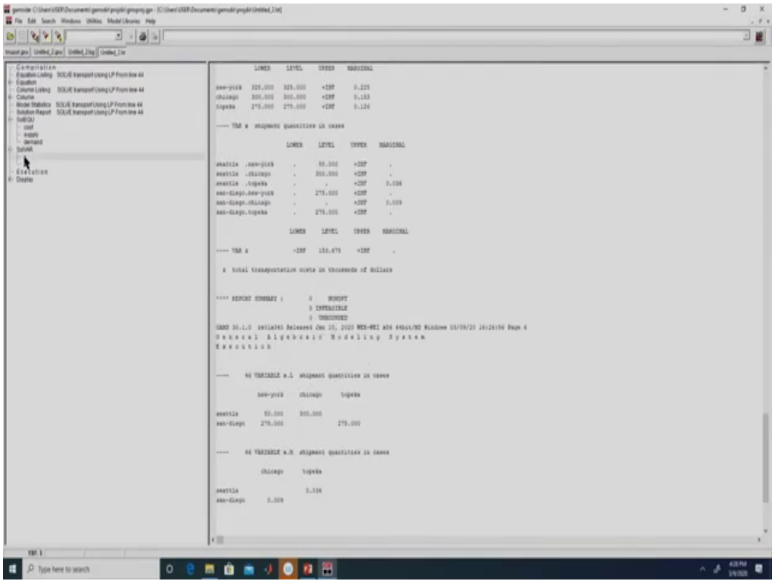
So, the listing file let me close this right.

(Refer Slide Time: 51:13)



So, the listing file as we had shown you earlier as this left hand side which is more like an index and the file itself is on the right hand side this entire thing is the listing file. So, over here we can go and click on anything. So, for example, let us say we are not interested in analyzing the entire list file, we want to see only the values of the variable x.

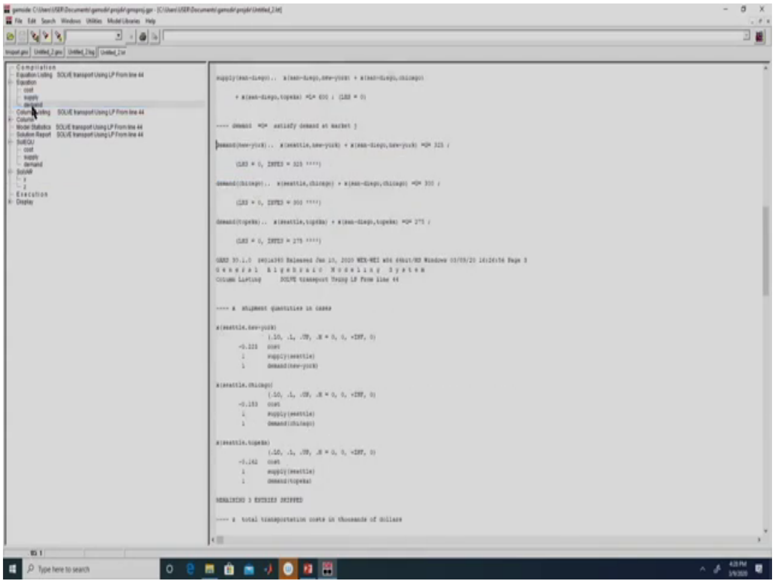
(Refer Slide Time: 51:32)



So, we can double click on this and it will correspondingly show only that particular value.

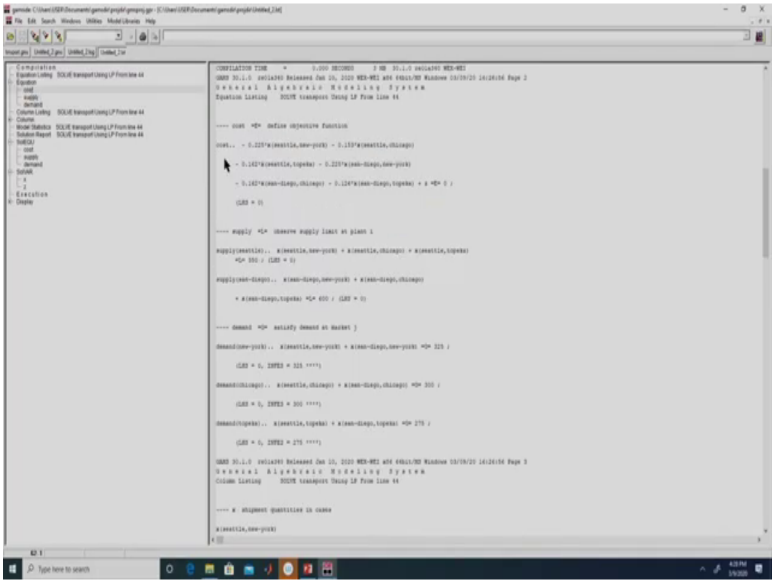
If you want to look at the equation listing let us say you want to look only at the demand equations listing.

(Refer Slide Time: 51:43)



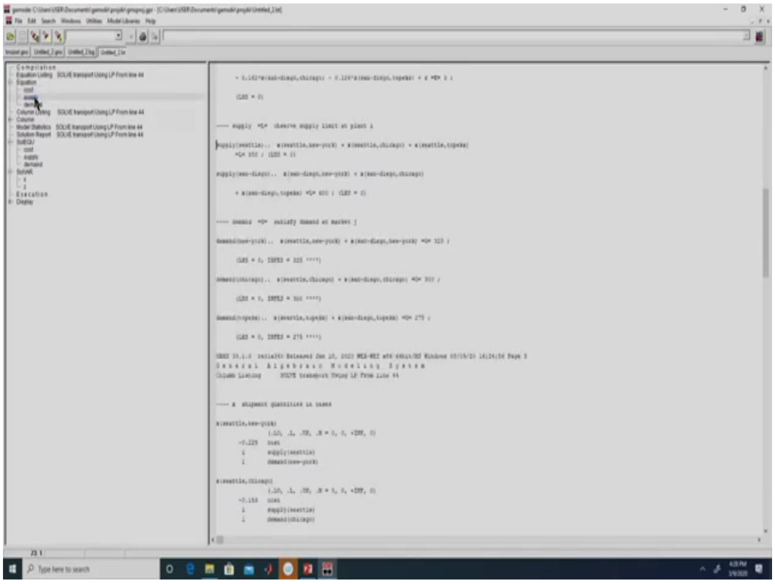
So, it directly goes over here right.

(Refer Slide Time: 51:47)



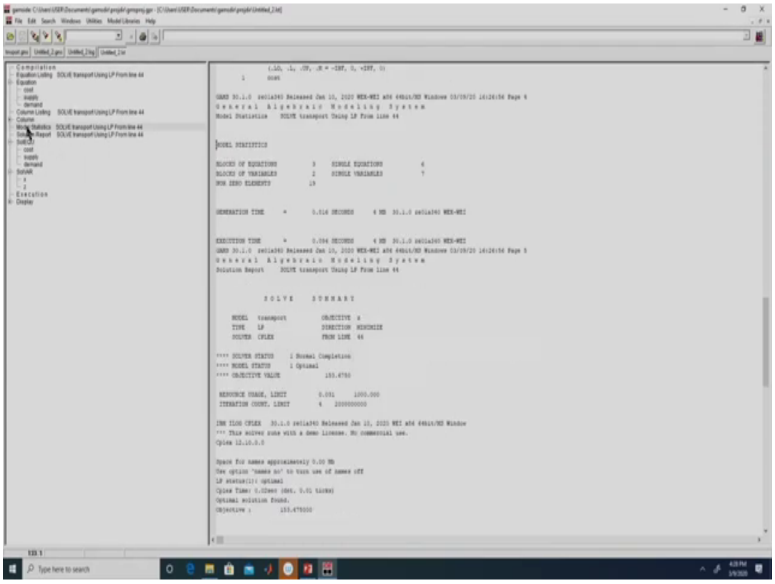
If you want cost, then double click on this similarly the supply.

(Refer Slide Time: 51:52)



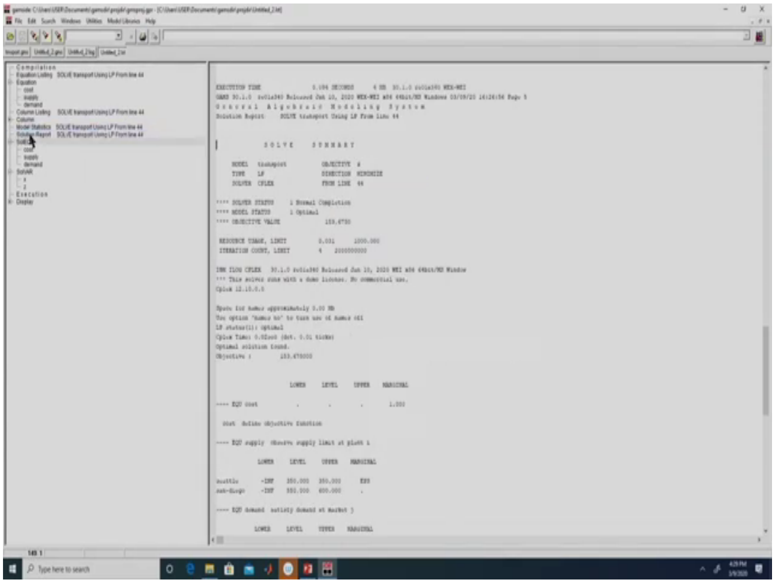
If you double click on this it will show the equation listing of the supply constraints.

(Refer Slide Time: 51:57)



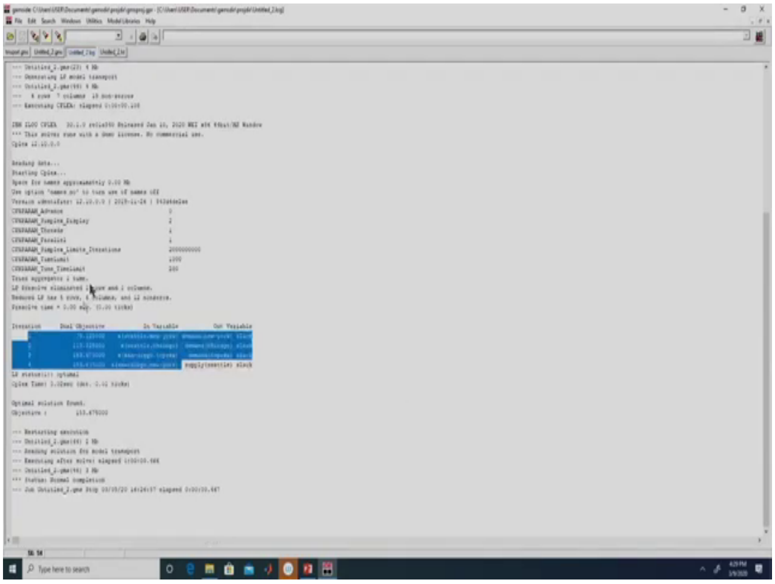
Over here we have the model statistics, which we had discussed and the solution report or the solid summary over here right.

(Refer Slide Time: 52:00)



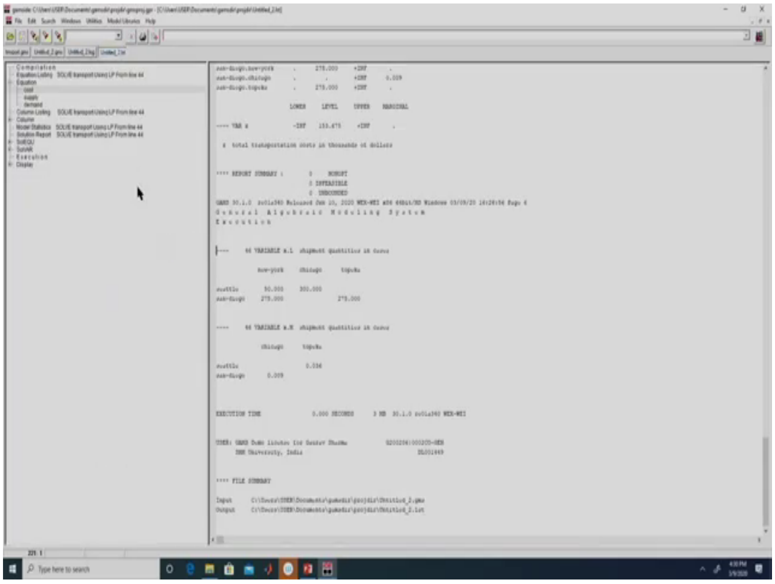
So, over here it directly specifies 4 iterations were utilized right.

(Refer Slide Time: 51:09)



The details of the iterations are available in this log file. So, over here we also have this display statement right display is the key word x dot l and x dot m right. So, when we specify this display x dot l and x dot m it gives the final values of the decision variable l and also the marginal prices associated with x we leave it to you to find out what is what are marginal values right.

(Refer Slide Time: 52:33)

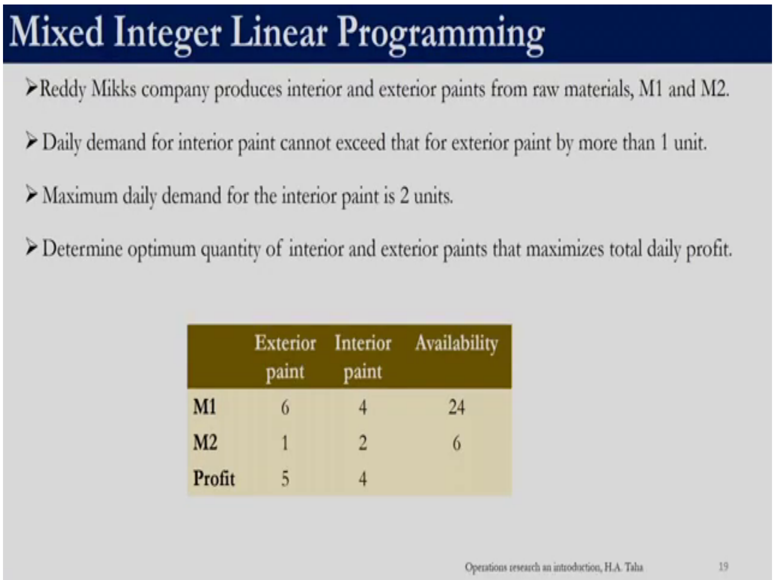


So, over here if we go to this display part we will have x over here.

So, this is x dot l which are the values which we discuss right that nothing is shipped from Seattle to Topeka; because it was 0 and nothing is being shipped from San Diego to Chicago. This part is being displayed only because of the display statement and these are the marginal prices.

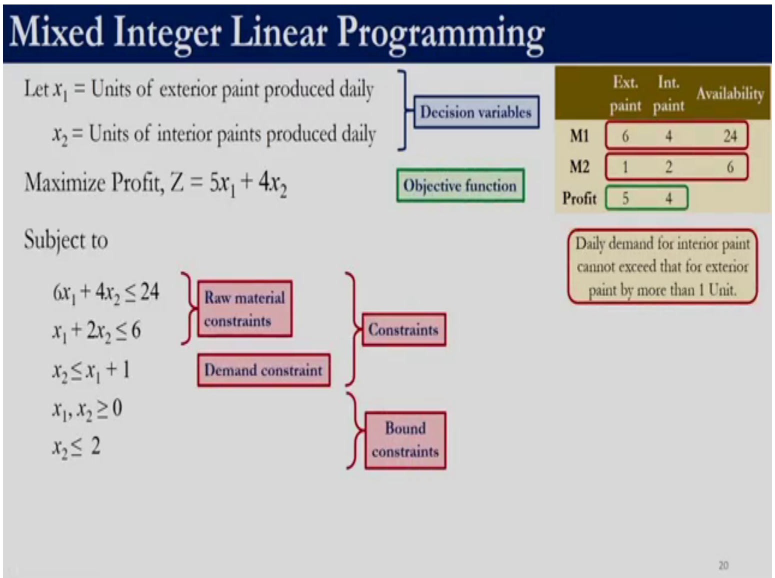
So, this example which we discuss was a linear programming problem right. Now let us look into a mixed integer linear programming problem. For mixed integer linear programming problem non-linear programming problem mixed integer non-linear programming problem; much of the content remains the same except for the solve statement in which we need to specify the appropriate category of the problem.

(Refer Slide Time: 53:18)

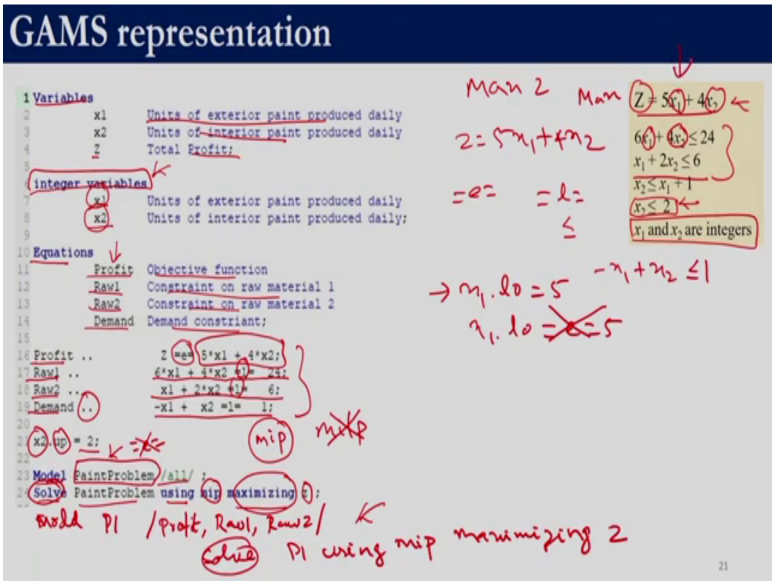


So, now let us look into a mixed integer linear programming this is the example of a paint company which we had discussed previously right.

(Refer Slide Time: 53:23)



(Refer Slide Time: 53:25)



So, we will not go into the problem this is the profit. So, our objective was to maximize the profit and it has two decision variable x 1 and x 2; in this case both of them are integer variables x 1 and x 2 right. So, what we will do over here is we will define variables.

So, in this case we have three variables; x 1 x 2 and Z; Z is also a variable remember in GAMS we need to specify variable right and the objective function is to be equated to that variable right. So, that is why we have this z over here we have description for all the three variables right.

And then we specify integer variables as x 1 and x 2. So, since we are listing x 1 and x 2 under this integer variable, it will be ensured that the values of x 1 and x 2 are integers over here we have 5 equations, but this is more like an upper bound. So, we will not treat it as an equation we will specify that the upper bound of variable x 2 is 2. So, basically we have three constraints and since GAMS requires the objective function to be an equation we have another equation over here.

So, basically we have 4 under equations we defined Profit Raw material 1 Raw material 2 and Demand; these are just names for the constraints right. And this is the corresponding description of those 4 constraints right over here we specify the constraint. So, Profit dot and then we will said Z is equal to e because this you can consider it to be an equality constraint 5 x 1 plus 4 x 2 is an equality constraint and then we are saying maximize z.

Remember e equal specifies a constraint whereas, if is just say z is equal to 5; that means, we are assigning a value of 5 to z or if we say z is equal to 5 x 1 plus 4 x 2, that would merely indicate that we are assigning the value of 5 x 1 plus 4 x 2 to be z right. If it is a constraint we need to give it in the form of equal to e equal to and then we specify the other two constraints right. So, over here it is 6 x 1 plus 4 x 2 should be less than or equal to 24.

So, 6 into x 1 plus 4 into x 2 is l equal to 24 equal to l equal to because the constraint that we have is less than equal 2. Similarly the second constraint is x 1 plus 2 x 2 is less than or equal to 6. So, x 1 plus 2 x 2 equal to l equal to 6 right. So, this is not 1 this is l. Similarly we have this demand constraint. So, the demand constraint we rewrite as minus x 1 plus x 2 should be less than or equal to 1.

So, over here minus x 1 plus x 2 is less than or equal to 1 and this upper bound we are not specifying it as an equation right. So, if we want to specify an upper bound, we can use the name of the variable in this case x 2 dot up. So, up will help us to specify the upper bound if we want to specify the lower bound then we can do x 1 dot l o is equal to 5.

So, this will specify the lower bound for the variable x 1. So, X 1 cannot take a value lower than 5. So, in this case we are specifying x 2 dot up is equal to 2. Remember over here it is not equal to e equal to because this is not a constraint that we are writing right here we are just specifying that the upper bound is 2 or. So, we should not use equal to e equal to we just say x 2 dot up is equal to 2.

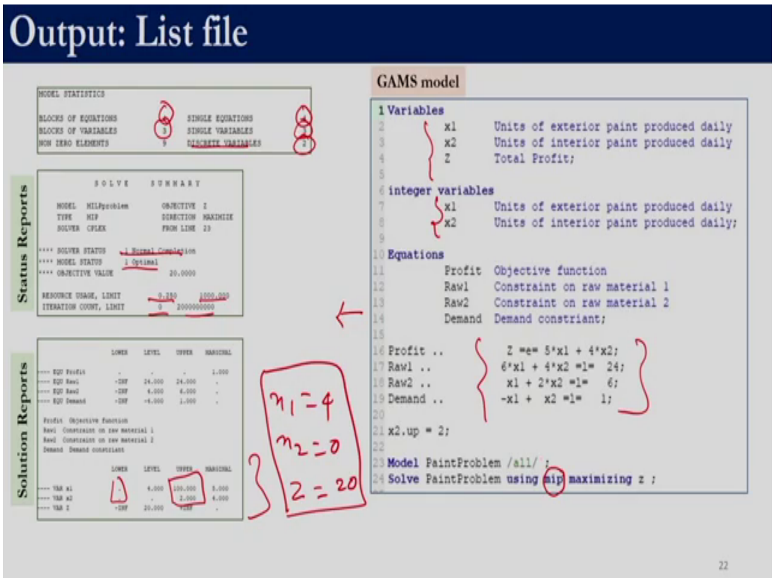
Similarly, if the lower bound is 5 we just specify x 1 dot lo equal to 5 and not x 1 dot lo equal to e equal to 5. So, this is not to be done. So, over here we want to include all these 4 equations. So, between this slash operator we have this all and we are construction a model whose name is paint problem.

Then we are solving the same model which we built in solve paint problem using right not milp, but mip right. Remember GAMS does not recognize milp, it only recognize mip, but it basically refers to mixed integer linear programming. And in this case we want to maximize right. So, the previous problem we had minimizing because we wanted to minimize the variable said over there in this problem we want to maximize right. So, over here maximizing and then the variable right the variable z; because it indicates this equation which is what is our objective function.

So, over here if we want to solve the problem with demand constraint and without demand constraint and then compare the results right. So, here then we can say model right let us say P 1 P 1 is the name of the model. So, here we can give Profit because we want to include that equation Raw 1, Raw 2 and that is it we do not want to have this demand equation.

And then we can have another statements let us say solve P 1 using mip; because it will be a mixed integer linear programming problem maximizing the variable z. So, now, since we have two solve statement we will get results with respect to both the models; then we will be able to see the impact of this constraint on the profit. So, this is the GAMS model file for this problem and this is a section of the list file for this GAMS model right.

(Refer Slide Time: 58:34)

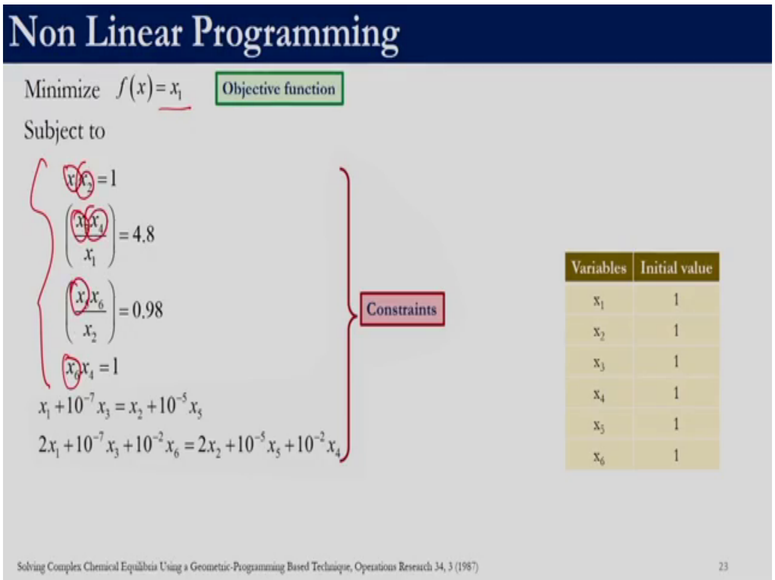


So, the status report shows Normal Completion and Optimal the resource that is used is less than the maximum right. And over here we have 4 blocks of equation and the number of single equation is also 4; because we do not have any set on which these equations are running over. And we have only 3 blocks of variable and each of the variable is actually a single variable over here. And over here we have 2 discrete variables right x 1 and x 2 are discrete variable. So, we have 2 over here.

So, this section provides us with the solution. So, the lower bound for the integer variables are 0 and the upper bound based on the constraint is hundred and 2 right. For the second variable we had specified that upper bound is 2; for the first variable the upper bound because of this equations comes down to 100. The final solution the optimal solution has a value of x 1 is equal to 4 and x 2 is equal to 0 right with the objective function value of z is equal to 20 right.

So, this is how we solve a mixed integer problem or mixed integer linear programming. So, now, we have seen how to solve a linear programming problem and a mixed integer linear programming problem. We will now look into how to solve a non-linear programming problem right.

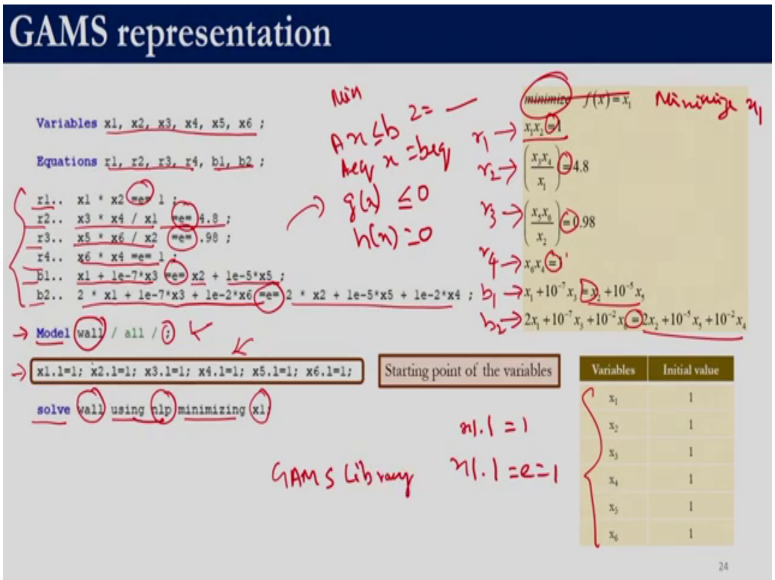
(Refer Slide Time: 59:56)



So, this is an example of a non-linear programming problem right. So, we want to minimize x 1 right, subject to these constraints.

So, we have 6 variables x 1, x 2, x 3, x 4, x 5 and x 6.

(Refer Slide Time: 60:10)



So, for that problem we do not need to define any additional variable previously we were defining z is equal to the objective function here the variable x 1itself is an objective function. So, basically what we can say is remove this and just say minimize x 1 right. So, in this case we have 6 variables. So, we define x 1 x 2 x 3 x 4 x 5 x 6 we have equation 1, equation 2, equation 3, 4, 5 and 6 right.

So, we have 6 equations let us name those equations as r 1, r 2, r 3, r 4 and b 1 and b 2 right. So, this is b 1 this is b 2 this is r 1, r 2, r 3 and r 4 right. So, we are specifying the equation r 1 dot dot space x 1 into x 2 equal to e equal to 1; because it is an equality constraint.

Second equation r 2 x 3 into x 4 divided by x 1 is equal to 4.8 right again equality constraint. r 3 is x 5 into x 6 by x 2 equal to e equal to because these are equality constraint. The 4th equation x 4 into x 6 is e equal to 1 right because of the equality constraint then equation b 1 right. So, x 1 x 1 into 10 power minus 7. So, here we write 1 e minus 7 into x 3 again equality constraints. So, we have equality e equality on the right hand side we have x 2 plus 10 power minus 5 x 5.

So, x 2 plus 10 power minus 5 x 5 right; similarly we have equation b 2. So, over here if you see unlike in MATLAB; wherein we were forced to provide the right hand side should be less than or equal to 0 for non-linear constraints and for non-linear equality it is of this form. So, we had to transform the problem into this particular form here there is no need to do anything of that sort right.

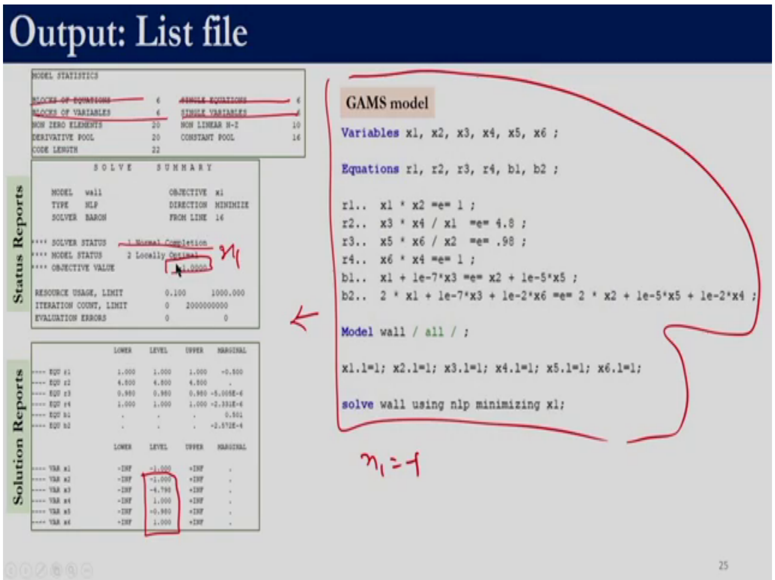
So, the right hand side can actually be an expression, it need not be a constant right. And again in MATLAB, we were restricted to solve with a minimization problem only here we do not have that problem. We can solve the maximization problem directly without transforming it into a minimization problem right. So, now, we are constructing this Model using the keyboard model. So, the name of the model is given us Wall right and we want to include all these 6 constraints over here right.

So, slash all slash right and we end it with a semicolon right. So, this will construct a model known as wall which will have all these 6 equations right. And over in this case we also specify the initial starting point. So, the starting point can be given as name of the variable x 1 dot l l means level equal to whatever that value is right. So, again remember it is not x 1 dot l equal to e equal to 1 right.

So, that is only for the constraint this is not a constraint this is just a starting point right. So, for all the 6 variables we are starting with a value of 1 right. So, that starting point is given and then we specify solve the model wall using nlp right; because this problem if you see there are no integer variables. But there are non-linear constraints nlp minimizing because its a minimization problem and we want to minimize the variable x 1 right and we end it with a semicolon right.

This model is also taken from the GAMS Library right. So, if you look into this model wall you will be able to get this code over there.

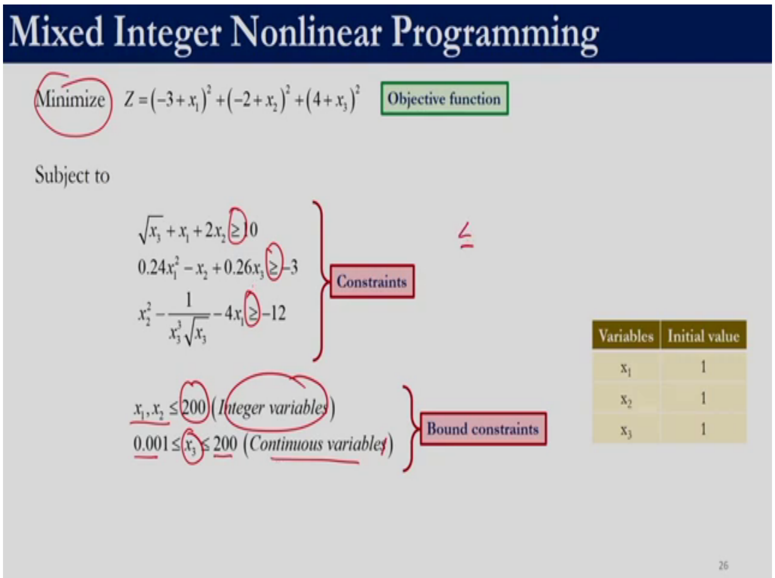
(Refer Slide Time: 63:43)



So, this is the solution for this model right. So, section of the list file is shown over here we are not showing you the echo print or the equation listing we are directly looking into the results. And we have Normal Completion the solver status is normal completion and the objective function value that we get is minus 1.

So, x 1 is equal to minus 1 right and the rest of the values can be obtained from here right; in this case x 1 was the objective. So, this value actually corresponds to this x 1 otherwise it just corresponds to the objective function value. Now we have seen a linear programming problem non-linear programming problem a mixed integer linear programming problem.

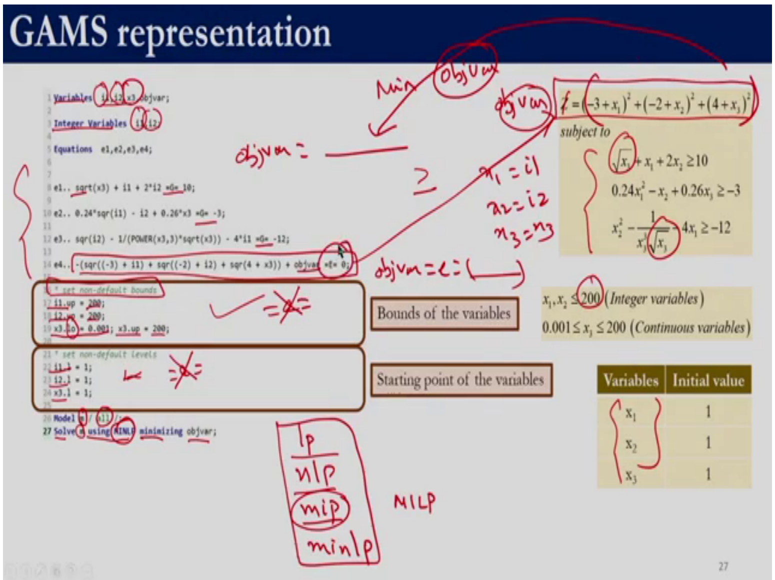
(Refer Slide Time: 64:28)



Now, we look into a mixed integer non-linear programming problem.

So, this is the example that we will consider for mixed integer non-linear programming problem. Here we have a minimization problem and the variables x 1 and x 2 are integer variables and they have an upper bound of 200; the variable x 3 has a lower bound of 0.001 and an upper bound of 200 and it is a continuous variable. And we have three constraints over here, all the three constraints are of this greater than or equal to nature. As discussed earlier we do not need to transform it into less than or equal to form.

(Refer Slide Time: 64:55)



So, since we have three variables first we are defining three variables which we already have in the problem. And we are defining another variable obj var right. So, this is obj var is equal to this and then we are saying minimize obj var this is the name of the variable right; instead of z we are using obj var again this model is available as part of GAMS library.

So, over here we have two variables which are integers right. So, the first variable i 1 and the second variable i 2 are integers. So, i 1 comma i 2 and semicolon over here we have three equations and as usual GAMS requires the objective function to be given as an equality constraint right. So, we have 3 plus 1, 4 equations. So, the 4 equations are given over here. So, this is sqrt helps us to take this square root. So, this equal to G equal to is because the constraints that we have are of the type greater than or equal to right. And over here we have the upper bounds for variable 1, 2 as well as the variable x 3 right.

So, what we have x 1 and x 2 over here, x 1 as become i 1 right to indicate that it is an integer variable x 2 as become i 2 to indicate it as an integer variable and x 3 what we have in the problem is maintained as such as x 3. So, over here we specify the upper bounds of the three variables for the variable x 3, the lower bound is 0.001 right. So, that is why we have x 3 dot lo. So, this lo stands for lower bound over here if you see we have this star over here. So, when we begin a line with a star it basically means its a comment right.

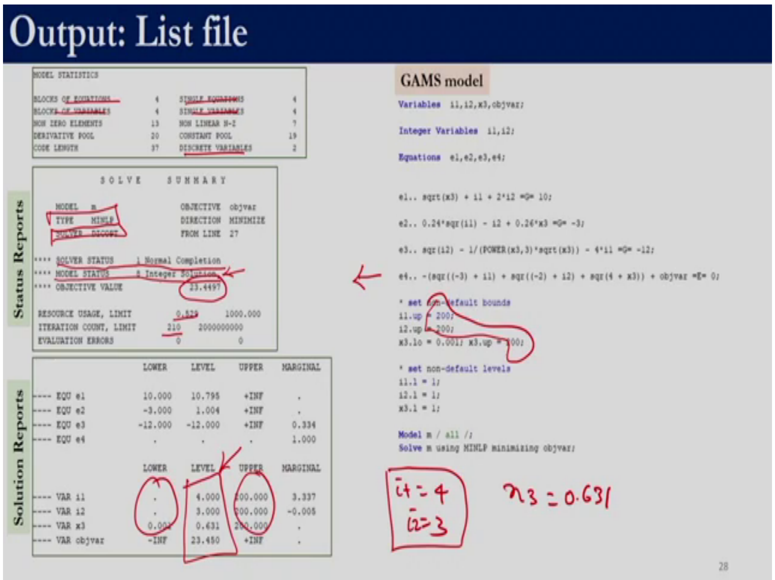
So, here we are specifying the bounds of the variable and in this case we have been given the initial value. So, we specify i 1 dot l is equal to 1 i 2 dot l is equal to 1 x 3 dot l is equal to 1. Again remember not equal to e equal to these are not constraints this is just a starting point; again for bounds we are not supposed to give it in terms of equality constraint.

So, we are constructing this model whose name is m and we want to include all the equations specified in this file. So, we have model m slash all slash semi colon and then over here we are solving using the keyword Solve we are solving the model m using MINLP right. So, this is again a key word. So, the key words which we have seen is lp nlp mip and minlp right

When we say mip it is actually a MILP right, but if you write MILP you will get an error you will have to write mip. So, that you will have to be careful depending upon the problem that we have at hand we will have to use one of these. And over here we want to minimize this objective function right. So, minimizing objvar because that is the variable which is defining the objective function right.

So, over here if you see objvar is over here right. So, what has been done over here is objvar is equal to what ever here we have right and then it has been taken to the left hand side. So, that is why we have this 4th equation. So, this basically corresponds to the objective function. So, we could have also written this as instead of this we could have just return objvar equal to e equal to and this expression whatever is given to us. So, we could have also done or we could bring it to the other side and give it as a equal to e equal to 0.

(Refer Slide Time: 68:24)



So, the solution of this GAMS model is given over here. So, the interpretation is similar the number of DISCRETE VARIABLES the number of EQUATIONS and variable BLOCKS OF EQUATIONS BLOCKS OF VARIABLES. Over here we have the SOLVER STATUS which is one Normal Completion the MODEL STATUS is 8 Integer Solution; this basically means that the solution which has been reported by GAMS is an integer solutions. Because we have some variables which are integers this gives the amount of time and the number of iterations it took to complete this problem right and this is the objective function value right.

So, over here values of the decision variables are given. So, this is the lower bounds these are the upper bounds which we had specified over here as 200. And these are the values which have been determined by the solver DICOPT right and the problem that we solved is MINLP right. So, here the decision variables are i 1 is equal to 4 i 2 is equal to 3 and x 3 is equal to 0.631 right.

So, the problem required i 1 and i 2 to be integer and over here we get integer values for that. So, this is how GAMS can be used to solve linear programming non-linear programming mixed integer linear programming and mixed integer non-linear programming problems right. Unlike MATLAB which was not able to solve mixed integer non-linear programming problems, which involved equality constraint; GAMS can solve those type of MINLP problems also. So, with that we will conclude this session.

Thank you.