Module 6: Solving LP Models in EXCEL with the SOLVER

Reading Material: 6.1 – Introduction

The previous modules have introduced the concepts of LP models. These models were small (two decision variables), so we could learn to set up the model and find the solutions graphically all in one motion. Obviously, few useful decision situations where LP is applicable have only two decision variables. So, we need a solution tool to truly help us harness the power of LP.

It is worth noting again that our effort to learn the algebraic fundamentals of LP is not just a network time-killer. History has shown (that's my long history by the way!) that it really helps our model construction skills and then ultimately our understanding of the EXCEL Solver as we move forward, expanding our modeling repertoire and gaining insight into model solutions.

6.1.1 The SOLVER and the Simplex Algorithm

This module introduces the EXCEL Solver for solving mathematical programming (e.g., LP) models. Solver utilizes the simplex algorithm to find the optimal solution of an LP model.

The simplex algorithm is an interesting "brute force" algorithm. It is based on linear algebra and employs an iterative approach, searching feasible region extreme points until it (the algorithm) can no longer find one better that the previous examined extreme point.

George Dantzig, the father of LP is given credit for developing this algorithm as a graduate student, and there is even a little bit of an urban legend surrounding its discovery. I won't bore you with the cute story (at least not here, probably in class though!). We will not further discuss simplex, but I would be more than happy to provide additional details to those who may be interested. It really is just an innovative use of algebra, a systematic analysis for improving solutions and conceptually mimics in a broad sense the graphical solution approach. It is my opinion that for most (all of you), if you understand how to solve a LP model graphically, you have mastered enough of the guts of the simplex process to satisfy me, and we can move on to bigger and better things without risk of not having enough underlying theory.

6.1. 2 EXCEL as a Platform for the Solver

EXCEL is just our calculator platform for implementing linear programming models. The Solver is an add-in to EXCEL (much like other add-ins you no doubt have used). Within Solver, the model builder will specify spreadsheet cell locations of model decision variables, the objective to be maximized or minimized, and the cell locations of the constraints.

After the modeler enters this definition, the Solver sets up the appropriate algebraic LP model and solves it (using Simplex), calculating an optimal solution and providing us the opportunity to generate the previously mentioned SA report, which can provide further insight about our decision situation.

EXCEL was originally designed strictly as an accounting spreadsheet (many, many years ago). Since then, many functions and add-ins have been added atop the basic intended functionality, and Solver is an add-in that should come with every installation of EXCEL. Frontline Systems developed the Solver, and their website (www.solver.com) has more information about the company and the Solver itself and more extensive (and expensive) versions of the Solver.

Note that there are many other specialized optimization tools that solve mathematical programming models. There are some pretty amazing tools. I have used LINDO and CPLEX (and probably other packages) in previous classes, primarily back in the early days just after PCs were introduced into the business curriculum (late 1980s, early 1990s). These modeling languages typically require input in EXACT algebraic format. Some offer programming tools to supplement the development of large-scale models. They are extremely powerful and useful tools, but can be expensive and require a steep learning curve. Because we are just learning about the potential of linear optimization, using some of these high-powered (and high-priced) tools for our class would be paramount to buying an expensive John Deere riding mower for lawn care of a 100 square foot back yard.

So, there are a couple of reasons why the EXCEL/Solver partnership is the right platform for the class. First, EXCEL is arguably the communication tool of the business world. Maybe it shouldn't be, but it is. People are always transmitting spreadsheets back and forth, and being able to further model an organization's data in the same milieu is a big benefit.

Second, and most important to me, in the old days, we had to input LP models into LINDO by typing in the algebraic model. One typo and LINDO would barf all over the place. Very remindful of my days accidentally leaving out ONE semi-colon in a PASCAL programming class assignment and setting the local record for most syntax errors in a single computer program at the University of Nebraska. Anyway, typing ability aside, a few of us may see the world in stark, algebraic form (because of age, learning style, whatever the reasons). But more of us may be creative and eventually find that we can immediately translate a word problem to a working LP model directly to EXCEL (no algebra in between!).

In the old days, you could not do this. With the EXCEL/Solver partnership, we can create LP models in either fashion (or frankly, using any kind of learning style or approach). I really think this flexibility makes the analytic tools in this class accessible to everyone. I have 20+ years of anecdotal (and empirical) data that would support this hypothesis.

Now, don't misinterpret my comments. We still need to get to the ONE correct LP model for our problems. My point is that we can just get there in different ways thanks to the flexibility of EXCEL and the Solver. We'll address this a little more in Module 9.

Even having said all this about flexibility, our starting point in using the Solver will be decidedly inflexible. We will simply mimic the algebra of an LP model in EXCEL (termed row/column format). Note that regardless of future demonstrations of cool, efficient ways of expressing LP models, THIS APPROACH in EXCEL (going back to the algebra) WILL ALWAYS BE ACCEPTABLE!

Different templates can be used to more easily create LP models as we begin to recognize different patterns in various business situations, Such a template is easiest, but there are always multiple ways in EXCEL in which the SAME LP Model can be implemented.

Put another way, the EXCEL representation from two different users for the same situation may seem very different, but the resultant algebraic LP model that the Solver ultimately sees and solves should be exactly the same. This flexibility is why spreadsheet modeling works so well for introducing the powerful world of mathematical modeling to all of you.

I also mention this because sometimes students get afraid to venture off in their own format, if you will, in this class. As long as the LP model that results is the same, venture away!

Reading Material: 6.2 – Finally, Introducing . . . the Solver!

After that long introduction, we are ready to illustrate the use of the EXCEL Solver. The three LEGO furniture production problems we previously looked at in Modules 3 and 5 as examples will be our first models.

6.2.1 The Assumptions

First, this module assumes that you, the reader, know the following:

1. How to do basic computation in EXCEL (adding, multiplying, etc.)
2. The distinction between absolute and relative cell addresses
3. The SUMPRODUCT and SUM function.

6.2.2 Row/Column Format

As previously mentioned, our initial Solver examples will simply mimic the LP model algebra by using a row/column format.

Spreadsheets are defined by rows and columns. Our analogy: An LP model has columns (decision variables) and rows (constraints, and the statement of the objective function). We will line up our model on the spreadsheet and the model coefficients (the numbers) will be our (only) inputs into the spreadsheet itself. We will also use a formula or two. Before we proceed, let us examine the Solver dialog box that we will use.

6.2.3 Solver Dialog Box Inputs

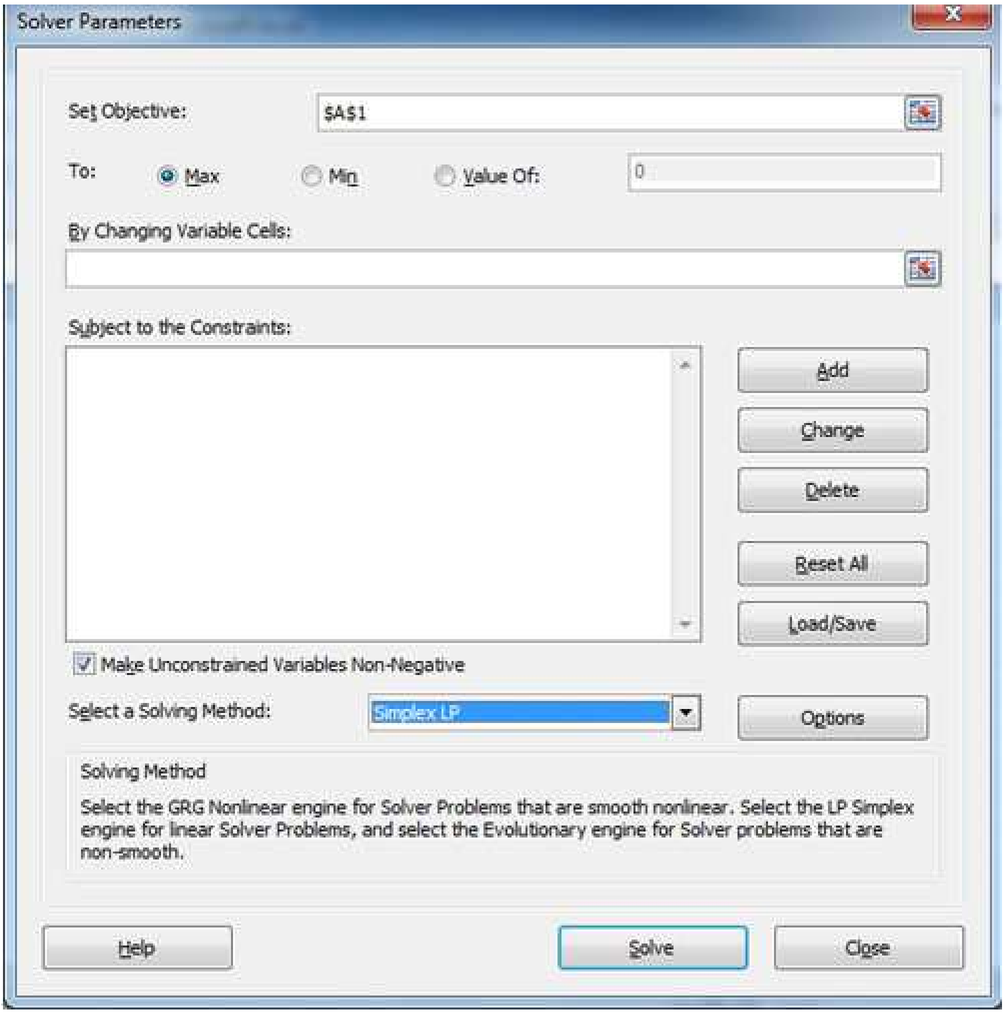


Figure 6.1

Figure 6.1 shows the dialog box for the Solver in Windows 2010. It is different from the dialog box for Windows 2007, but the basics are the same. Recall there are three components of every LP model: decision variables, objective function, and constraints. The Solver add-in is normally found under the Data Sub-Menu of EXCEL, in the Analysis area. If not present, follow the usual procedures for activating an EXCEL add-in.

The first item on the dialog box is the input area "Set Objective." This input is the single spreadsheet cell where the objective function value is being calculated. Why they have such a big space for input is beyond me. I would have designed it differently, but surprisingly, they didn't ask me. So, on our spreadsheet, we will have a "Target Cell" – the placeholder of our calculated criteria. The criteria will be maximized or minimized, and that also needs to be properly selected here in the Solver.

Next, the spreadsheet cell locations of our decision variables (they are referred to as changing cells here) are specified. We simply 'reserve' some cell space for our decision variables in our spreadsheet and the Solver will find their optimal values when the model is solved. It will place these optimal values in the reserved cells. Thus, all calculations in the spreadsheet for our objective function and for our constraints are linked back to these (blank, reserved) changing cells. They are linked by how we construct our formulas to create the algebraic model.

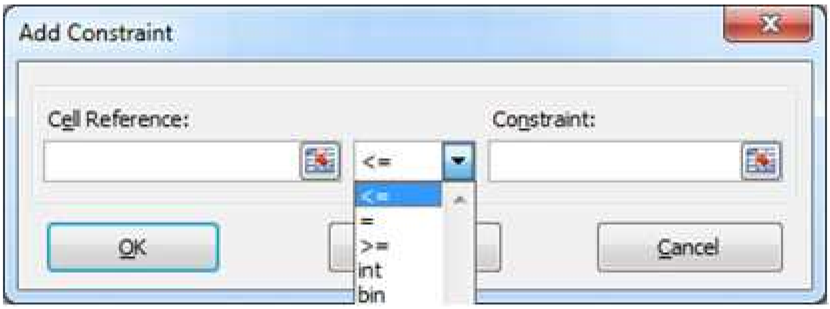


Figure 6.2

The third item in the dialog box of major importance is where we specify model constraints. Figure 6.2 shows the follow-up dialog box that pops up when we select add or modify a model constraint.

There are three components of the constraint pop up box – I don't really care for the headings used, but, as you perhaps could have guessed by now . . . they didn't ask me. Basically, I'd like to refer to them as the left-hand side (LHS) of the constraints (found on the left-hand side of the box), the right-hand side (RHS) of the constraints (found on the, yes, amazingly enough, the right-hand side of the box) and the item in the middle, the relationship between the LHS and the RHS.

In the row column form of our LP models, we will calculate the LHS of a constraint in a spreadsheet cell, and type the RHS value of a constraint (a number) in another spreadsheet cell. If our constraint was 2X + 3Y <= 30, a formula in one cell would perform the EXCEL calculation of 2X+3Y (call it cell J1), and the number 30 would be entered in another cell for the RHS values (call it Cell K1). Then, using the constraint dialog box, the first box would be "pointed" at J1, the second box at K1, the <= choice would be selected for the relationship, and our constraint would then be known by the Solver.

Note the six options for the "relationship choice." In due time, we will talk about the five most relevant ones (<=, >=, =, INT, BIN). You can probably guess what the first three represent, and maybe even the next two.

Finally, note the "Solve" button at the bottom of the dialog box. This is the magic button to hit after we have completely defined the LP model. A message will be displayed indicating whether we were successful – this is discussed later.

Two important miscellaneous items are worth mentioning that help smooth the model building and solving approach.

Remember the concept of non-negativity? In Windows 2010, the Solver defaults to ensuring our variables are non-negative. This is seen in the checkmark by the insightful phrase "make unconstrained variables non-negative." This is great progress – past versions of the Solver required a visit to the options page and forced you to check a box.

The second important item is the menu choice for “Select a solving method.” This entire class is based on the premise that you choose “Simplex LP.” Why is it not the first choice? Well, it should be, but never mind. If you don’t select Simplex LP, you are choosing a non-linear Solver that might get the correct optimal solution, but more than likely, especially as our problem sizes get bigger, it will return a suboptimal solution. The non-linear solvers are good for non-linear models. But non-linear solvers can never guarantee finding the optimal solution for any problem. Linear models solved with a linear solver are GUARENTEED to find the best solution. We will gain much more in solution power and model insight even if we have to simplify reality a bit to have a linear model. We previously talked about this trade-off back in Module 2. So, always choose Simplex LP Solver!

Finally, Figure 6.3 shows the options box that appears if you select “Options.” There is not much relevant here at present, but I wanted to let you know it exists in preparation for potential future visits.

6.2.4 Basic Steps to Prepare the Spreadsheet

We re-state here the preparatory steps to get our spreadsheet ready (i.e., format the LP model appropriately) for the Solver.

a) Highlight an area in EXCEL in which the solver will calculate the values of the decision variables. Typically I like to outline the cells, but that is just a habit of mine. Remember, we are mimicking the algebra of our LP models, so we can expect to see a row of decision variables, where each column corresponds to a specific decision variable.

b) We will add a row for our objective function calculation. Here, we will use the SUMPRODUCT function to relate the objective function coefficients (the numbers in this row) back to the (initially empty) decision variable cells. This will calculate the target cell, the “set objective to” cell.

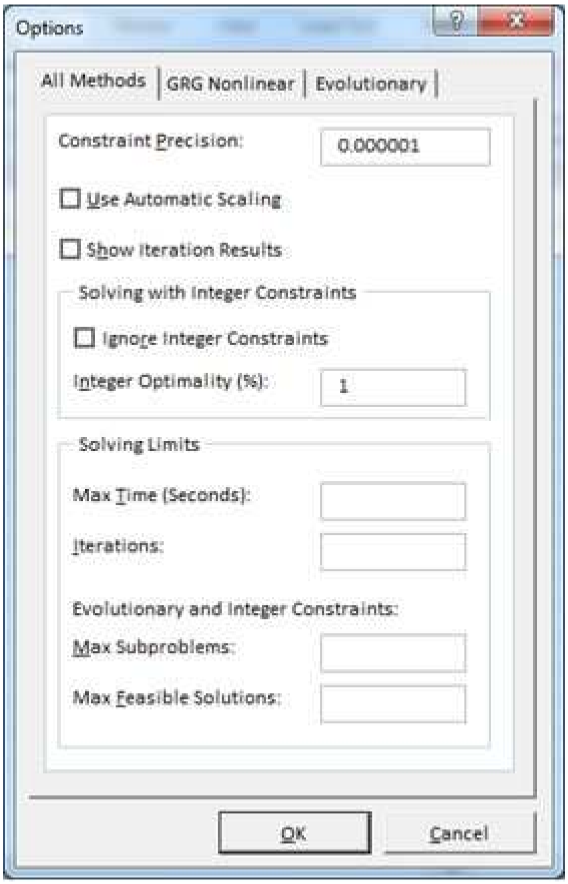


Figure 6.3

c) We will also add a row for each constraint. Similarly to the calculation of our objective function, the SUMPRODUCT function will calculate LHS (usage) of constraint by relating/linking the coefficients of the constraints that we place in the cells of our row back to the originally empty changing cells.

d) Finally, we enter a number for the RHS of our constraints. This prepares us for the need in Solver to relate the LHS to the RHS value to form model constraints.

Whew! That's a lot of talking, and not a lot of doing. But we have some solid checklists that we can fall back on to help us construct our LP models. The next three examples are meant to illustrate this process through screen shots. I encourage you to work these practice problems alongside the book to get the hang of it.

And remember, when all is said and done, we will not only find the optimal solution of the model, including the best achievement of our objective, but we will also be able to identify binding and non-binding constraints. Additionally, the Solver will also allow us to generate the SA printout, which can give us additional simple insight into the situation being modeled (discussed in Module 7).

Reading Material: 6.3 – Example 1: Basic LEGO Furniture Production

Back in Module 3, we created our first LP model to analyze the LEGO furniture production situation. The LP model constructed is repeated below. Note that TAB is the decision variable representing the optimal number of tables to produce and CHR the decision variable representing the optimal number of chairs. The first constraint dealt with the available number of small LEGO blocks, the second constraint was the available supply of large LEGO blocks, and the third constraint was a capacity constraint.

MAX	16 TAB + 9 CHR	
ST	2 TAB + 2 CHR	<= 400
	2 TAB + CHR	<= 300
	6 TAB + 4 CHR	<= 960
	TAB, CHR	>= 0

6.3.1 Creating the EXCEL Template for LEGO Production

Following the directions provided above, our task is to mimic the standard form of a LP model in EXCEL.

Figure 6.4 shows our initial template using the row/column format for this LP model. Cell C4 is the placeholder for our TAB decision variable. Cell D4 is the placeholder for our CHR decision variable. As is my custom, I have outlined the decision variable cells with a border. This is just an artistic touch; it has nothing to do with creating the correct LP model. (And yes this is about as artistic as I can get.)

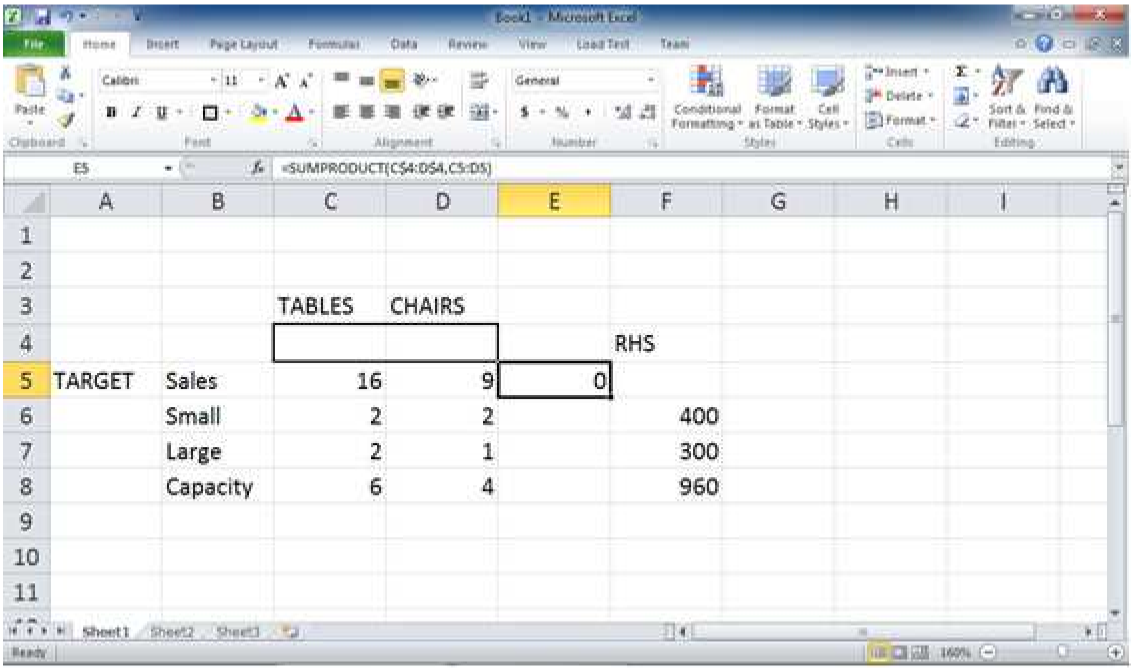


Figure 6.4

Row 5 is the objective function row, where the objective function coefficients (sales accrued per unit of furniture) are entered. Rows 6 through 8 are the holders of the usage values for the Small and Large LEGOs, and the capacity constraints. In all four rows, the correlation between the coefficients in the algebraic representation of the LP model above and the numerical values entered into the EXCEL spreadsheet should be apparent. We are simply mimicking the algebraic model.

Column F is the location of the RHS values of our constraints. Again, this information is entered into the spreadsheet.

Column E is an action column – for instance, related to sales, it ties together the coefficients entered with the placeholders for our decision variables and calculates the objective function value (our target cell, cell E5). Similarly, it is also where the LHS values of our constraints are calculated (cells E6 through E8).

Figure 6.4 also shows the entry of the first SUMPRODUCT formula needed for Column E. It is in the target cell, the objective function value cell. Notice the explicit entry:

=SUMPRODUCT(C$4:D$4,C5:D5)

This function aligns the two ranges, multiplies together the individually aligned cells, and adds up the individual products (it “sums the products,” thus the formula name of SUMPRODUCT). In EXCEL terms, this function is equivalent to:

=C$4*C5 + D$4*D5

This is, of course, the objective function, or, algebraically, 16*TAB + 9*CHR.

Note also the purposeful use of absolute cell addressing (use of the $). This allows us to type this formula one time, then copy it for the other actionable cells in Column E. Time saving, and error saving, if you type like me.

Figure 6.5 shows the template after the cell E5 formula is copied down to cells E6:E8. We have now completed the EXCEL set-up of our row/column LP representation of the LEGO production problem. The next step is to fill-in relevant Solver data.

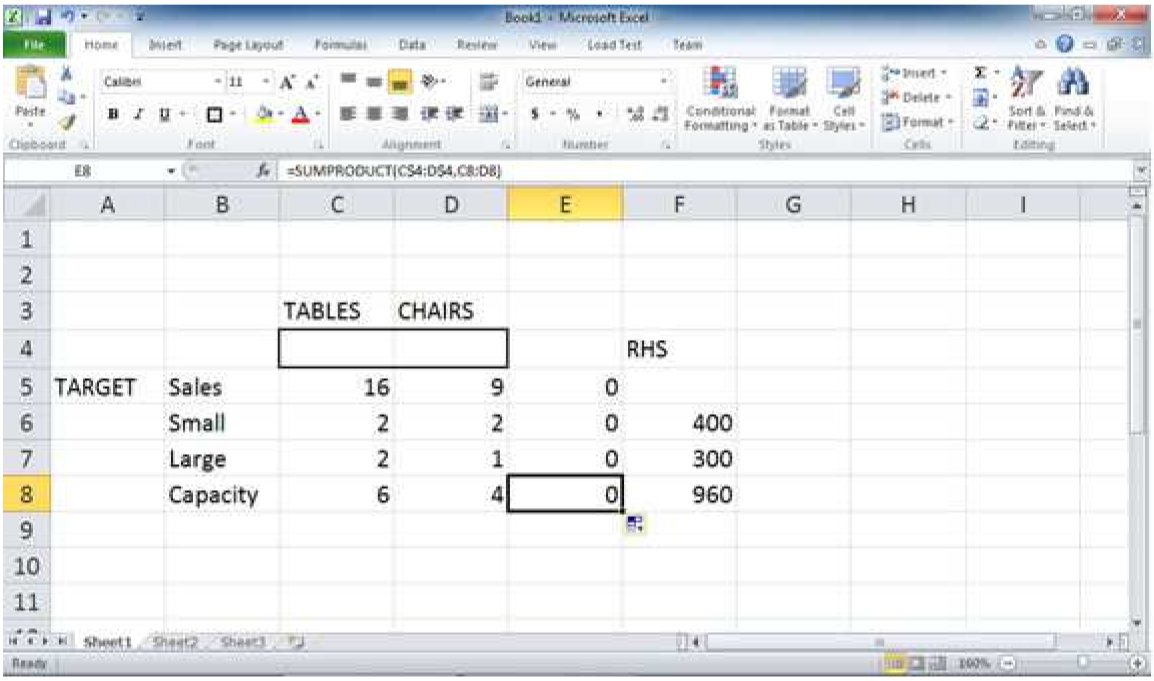


Figure 6.5

6.3.2 Identifying Model Components to the Solver Dialog Box

Figure 6.6 represents the final view of the Solver dialog box after all model parameters have been input.

First item – our target cell (set objective). We input the cell where the objective function is being calculated (E5), and identify that it is a MAX problem (by selecting the appropriate circle).

Next item – our decision variables (changing cells) – the cell placeholders that Solver will fill in with the optimal production levels. We input the range C4:D4. (You can experiment with this, but you can just highlight the area, you do not have to actually type the cell references).

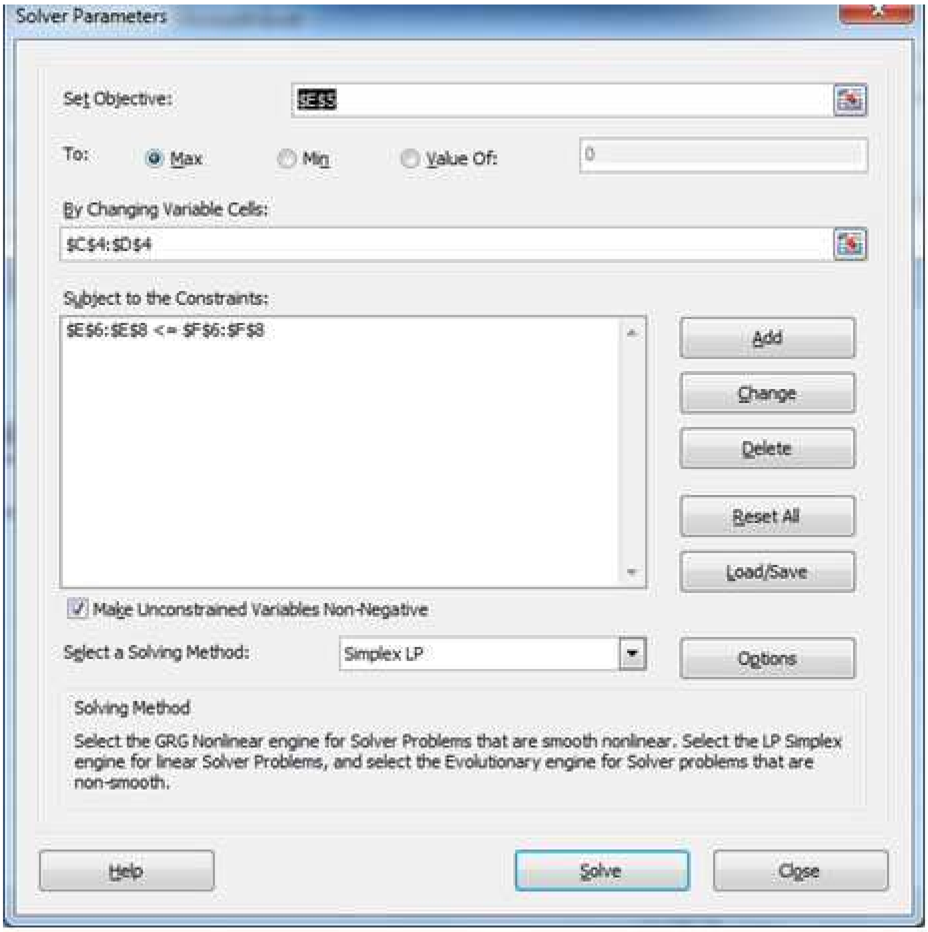


Figure 6.6

Final major item – identifying the constraints to the Solver. Now, I took a shortcut – I did all three constraints at the same time. I could have done them one at a time, but because all three constraints are the same "flavor" (less than or equal to), I could use the power of EXCEL and do it quickly and just once. Figure 6.7 is the dialog box used to add these constraints. I created the LHS range (E6:E8), picked the relationship (<=) and then created the RHS range (F6:F8). As before, I did not actually type the cell references, but highlighted them and EXCEL Solver did the rest.

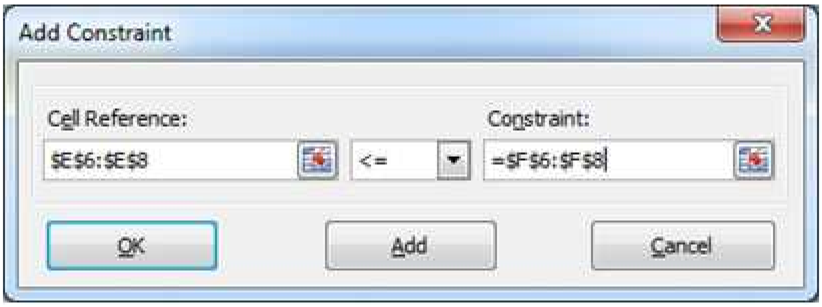


Figure 6.7

Double checking, the non-negativity box must be checked and the Simplex LP solver selected.

It appears we are ready to hit the “Solve” button.

6.3.3 Yes, Houston, We Have a Solution!

The next step cannot be overrated, especially as we are gaining confidence in this whole modeling/Solver thing. Figure 6.8 shows the response that one hopes to receive from the Solver. *Always read the blue box carefully before proceeding!* Your message should read “Solver found a solution. All constraints and optimality conditions are satisfied.”

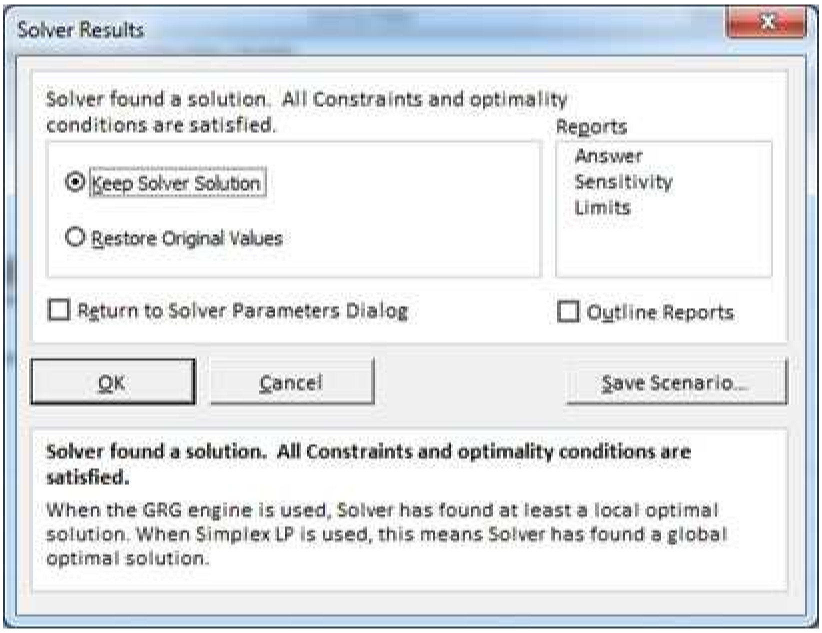


Figure 6.8

I wish it would flash lights and ring bells and sound off buzzers instead of just giving you this message. Why? Well, look at Figures 6.9 and 6.10. These are blue boxes that give you bad news. Figure 6.9 indicates your model is infeasible (meaning that the constraints are in conflict and there is no feasible region) and 6.10 indicates your solution is a Buzz Lightyear solution (to infinity and beyond! *Unbounded* is the more formal term), meaning that something is amiss.

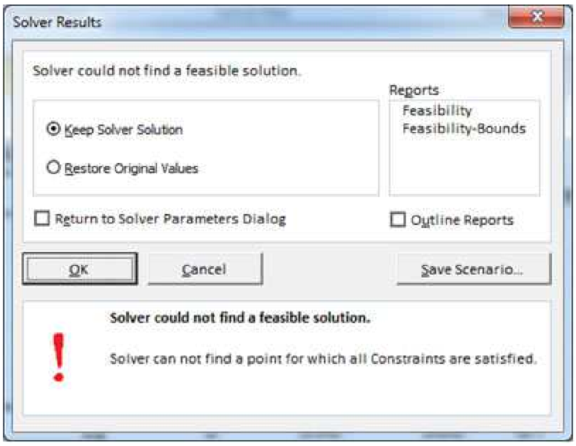


Figure 6.9

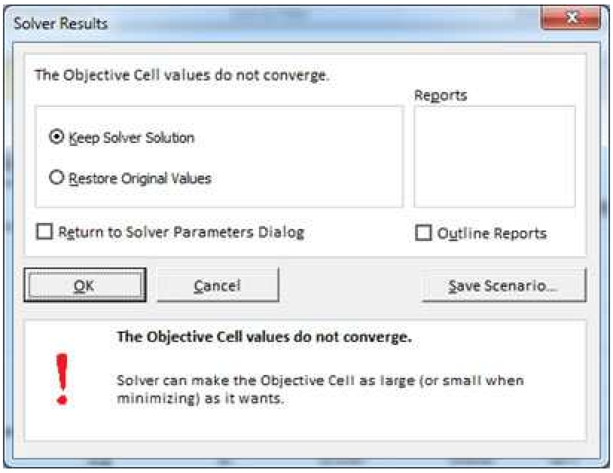


Figure 6.10

The bad thing – the Solver goes ahead and puts some random numbers in your changing cells, and if you don't read the message carefully you might be misled that you have a working LP model. So, my friends make sure you get the "Solver found a solution. All constraints and optimality conditions are satisfied" message before moving forward. It will save you a lot of pain and agony.

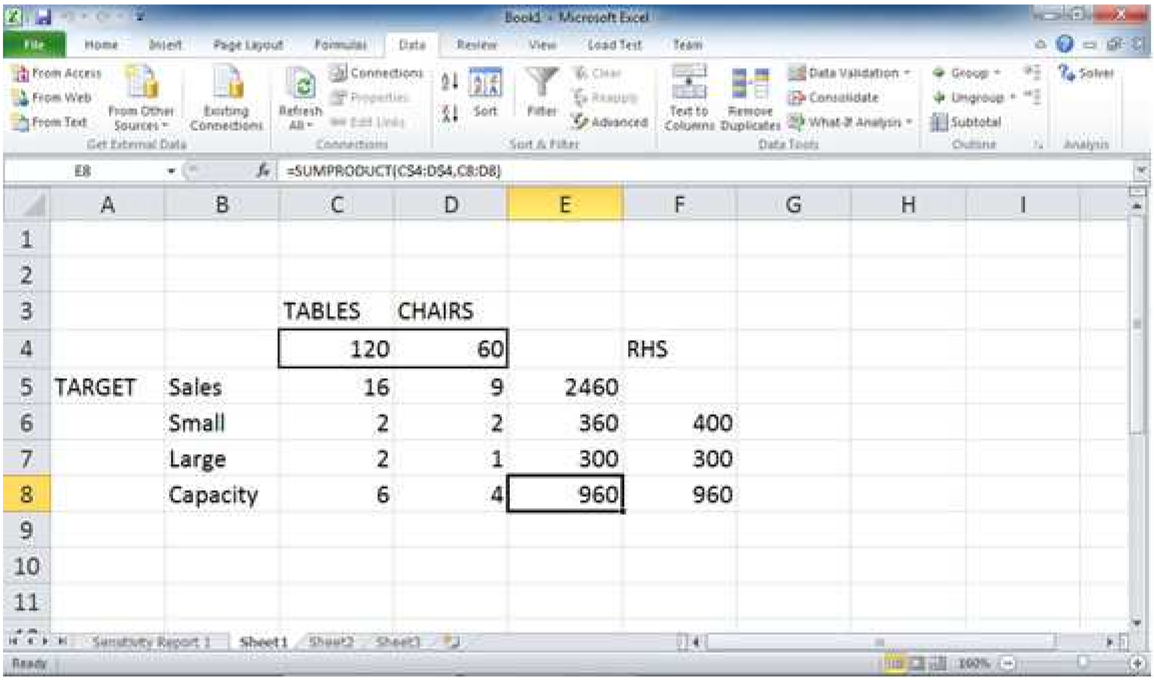
Returning to Figure 6.8, take a look at the RHS of the blue box. Under "Reports" we can highlight the SA report and Solver will automatically generate additional insightful data, added as a separate sheet in the workbook.

Note that there are two other reports that can be highlighted – the Answer Report and the Limits Report. If you are curious, go ahead and take a look at them. They are redundant and add no useful value to our modeling process. Thank goodness they are included.

Figure 6.11 shows the results after we received the good news ("Solver found a solution") from Solver. The changing cells have been populated with the determined optimal production mix (120 tables and 60 chairs) for total sales of $2460. This solution uses all of the large LEGO blocks (Row 7 LHS = RHS, 300 units) and all of the cu units of capacity (Row 8 LHS = RHS, 960 units). Of course, we already knew what to expect solution wise, but this confirms a successful completion our first LP model in EXCEL.

Figure 6.12 is the aforementioned SA report. We will study it more in the next Module.

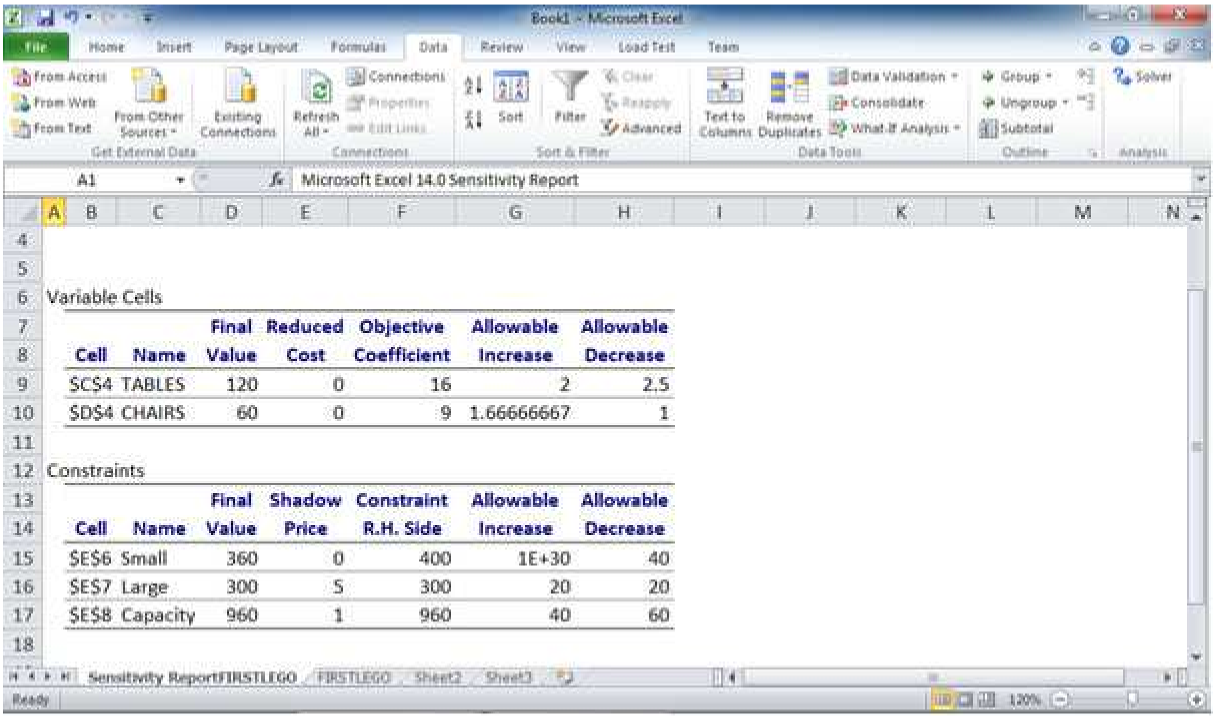
For completeness and more examples, the other two LEGO furniture examples (from Module 5) are shown next.



=SUMPRODUCT(C$4:D$4,C8:D8)

	A	B	C	D	E	F
3			TABLES	CHAIRS		
4			120	60		RHS
5	TARGET	Sales	16	9	2460	
6		Small	2	2	360	400
7		Large	2	1	300	300
8		Capacity	6	4	960	960

Figure 6.11



Microsoft Excel 14.0 Sensitivity Report

Variable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
C4	TABLES	120	0	16	2	2.5
D4	CHAIRS	60	0	9	1.66666667	1

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
E6	Small	360	0	400	1E+30	40
E7	Large	300	5	300	20	20
E8	Capacity	960	1	960	40	60

Figure 6.12

Reading Material: 6.4 – Example 2: The LEGO Furniture Production Scenario – with the Proportion Constraint

This model was previously defined in Module 5. Basically, a constraint that implements the production restriction that tables should be at least 80% of the total production combined was added to the original model. Module 5 discussed the variety of different ways that we could model this requirement. All of the following are possible implementations of the 80% requirement (i.e., they are all mathematically equivalent).

a. TAB >= .8 (TAB + CHR)

b. TAB >= .8 TAB + .8 CHR

c. .2 TAB >= .8 CHR

d. TAB >= 4 CHR

e. .2 TAB – .8 CHR >= 0

f. .8 CHR <= .2 TAB

g. –.2 TAB + .8 CHR <= 0

There are other candidates of equivalent constraints as well. In Module 5, we found version d) most useful. For EXCEL, either e) or g) fit row/column format – a number on the RHS (it happens to be 0, but 0 is still a number!) and decision variables on the LHS. Yes, the decision variables have negative coefficients, but that is just fine.

Now, we can implement all these different forms in EXCEL as well, but remember, for the time being, we are sticking to strictly a row/column template, so we will not vary from that presently.

Figure 6.13 shows the following EXCEL template for the following LEGO production model:

MAX 16 TAB + 9 CHR

ST 2 TAB + 2 CHR <= 400

2 TAB + CHR <= 300

6 TAB + 4 CHR <= 960

0.2 TAB – 0.8 CHR >= 0

TAB, CHR >= 0

I opted to use version e) of our proportion constraint in the model. I could have used g) and still been in row/column form. But this allows us to see a greater than or equal to constraint in the model and to show (if it wasn't clear) how multiple constraint families look in the Solver dialog box. The optimal solution for the LP model is also shown in Figure 6.13.

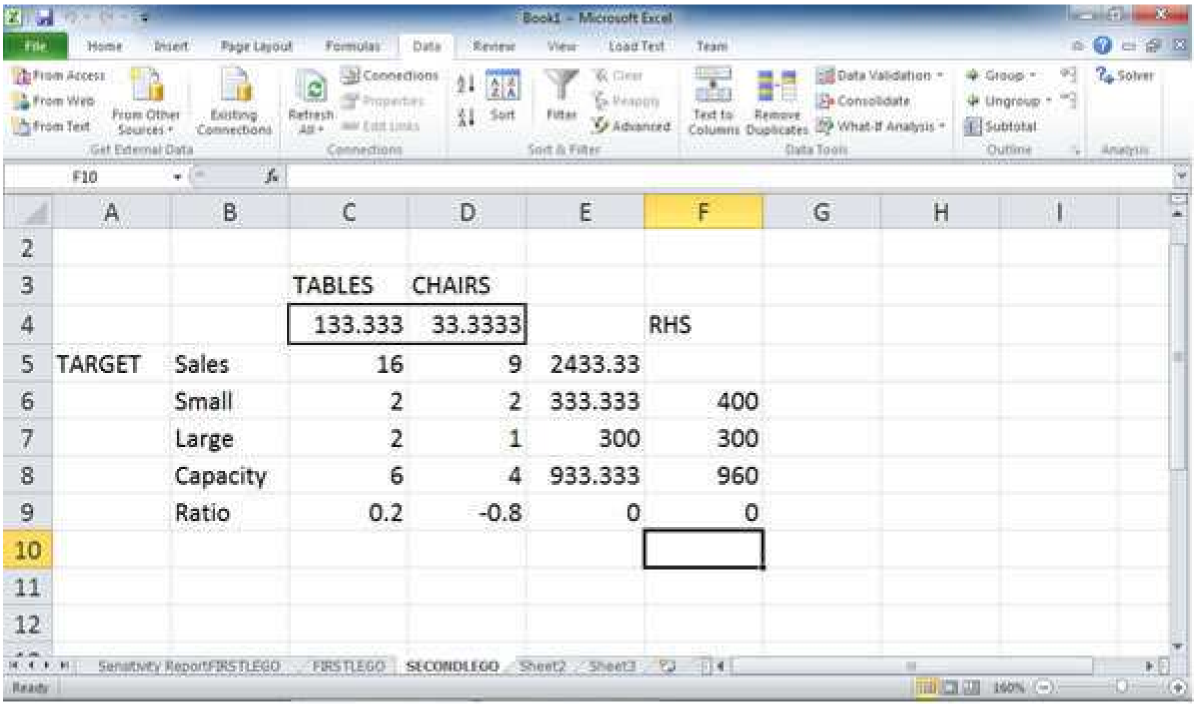


Figure 6.13

Figure 6.14 is the Solver dialog box for this model. There should be no surprises. As before, we are just mimicking the algebra of the model and Column E remains the only column where any calculations occur (through the use of SUMPRODUCT). We added our one additional constraint in Row 9, and because it is a greater than or equal to constraint, we HAD to make it separate from the other three less than or equal to constraints in Rows 6 to 8. Everything else in the model was exactly the same as our first example.

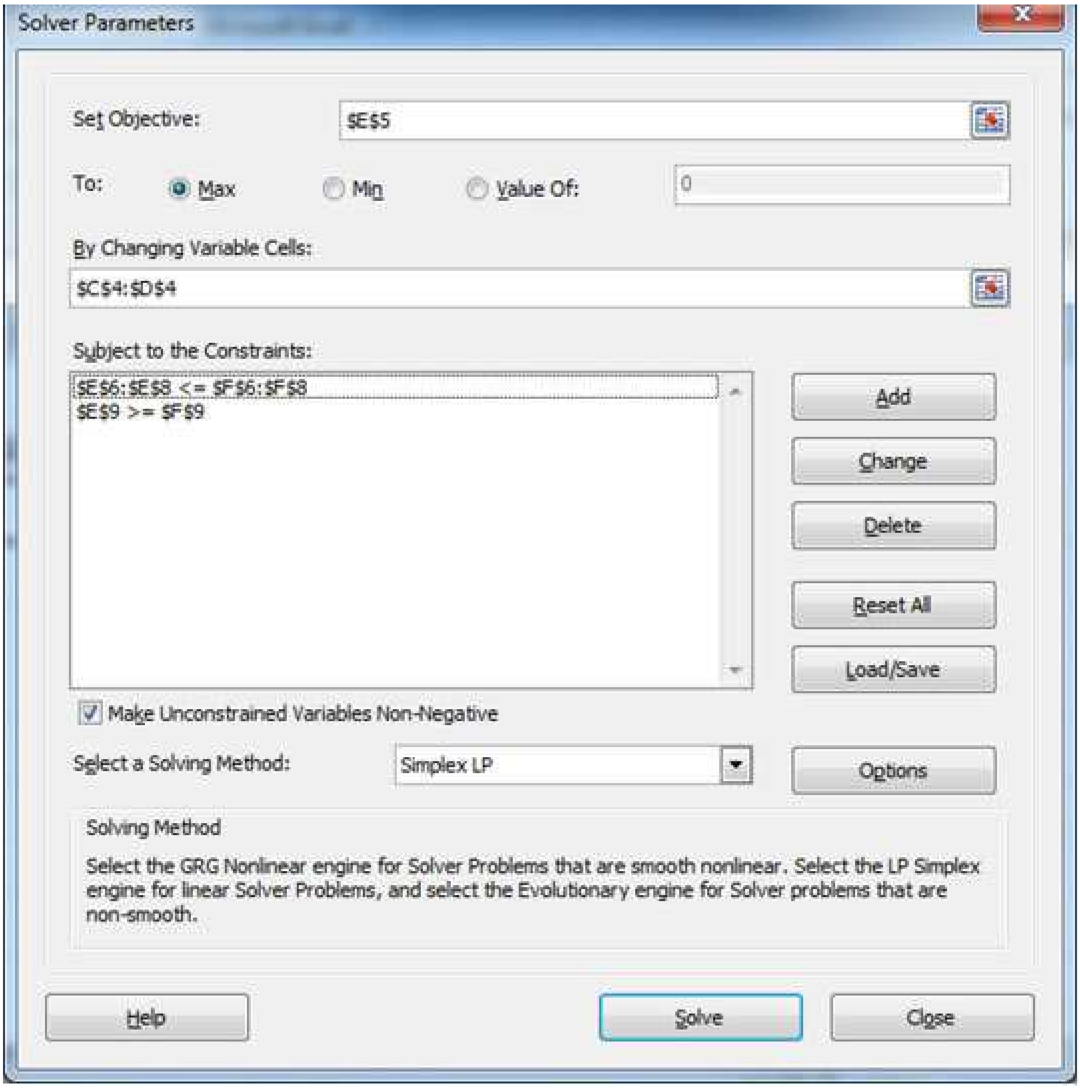


Figure 6.14

Reading Material: 6.5 – Example 3: The Minimization Version of the LEGO Furniture Problem.

Finally, for completeness, we include the spreadsheet (Figure 6.15), the Solver dialog box (Figure 6.16) and the SA printout (Figure 6.17) for the minimization version of LEGO furniture.

The LP Model from Module 5:

MIN 16 TAB + 10 CHR

ST:

TAB +	CHR	<= 175
	CHR	>= 60

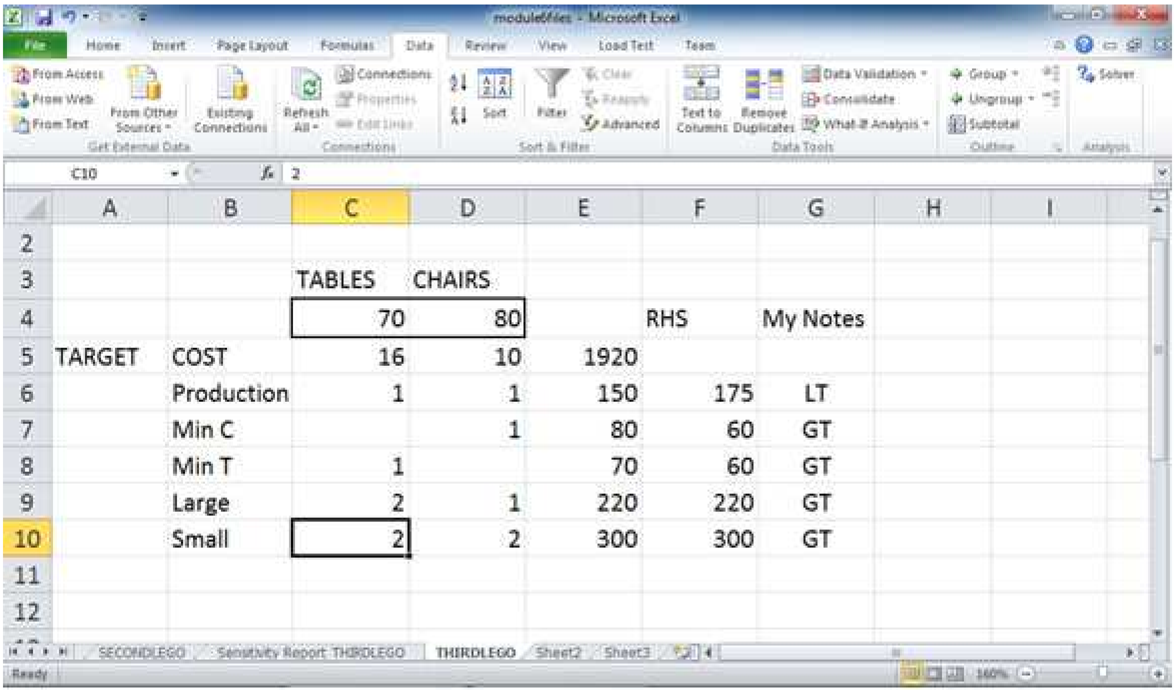
TAB >= 60

2 TAB + CHR >= 220

2 TAB + 2 CHR >= 300

TAB, CHR >= 0

Row 4 is the changing cells placeholder. Column E has the SUMPRODUCT formulas with the absolute cell references that link the changing cells (Row 4) to the coefficients in either the objective function value calculation (Row 5) or the five constraints (Rows 6 to 10). Column G is just a modeler's note on the flavor of the corresponding constraints in each row. It has no impact on anything. The optimal production levels are also shown in the spreadsheet in Figure 6.16.



	A	B	C	D	E	F	G	H	I
2									
3			TABLES	CHAIRS					
4			70	80		RHS	My Notes		
5	TARGET	COST	16	10	1920				
6		Production	1	1	150	175	LT		
7		Min C		1	80	60	GT		
8		Min T	1		70	60	GT		
9		Large	2	1	220	220	GT		
10		Small	2	2	300	300	GT		
11									
12									

Figure 6.15

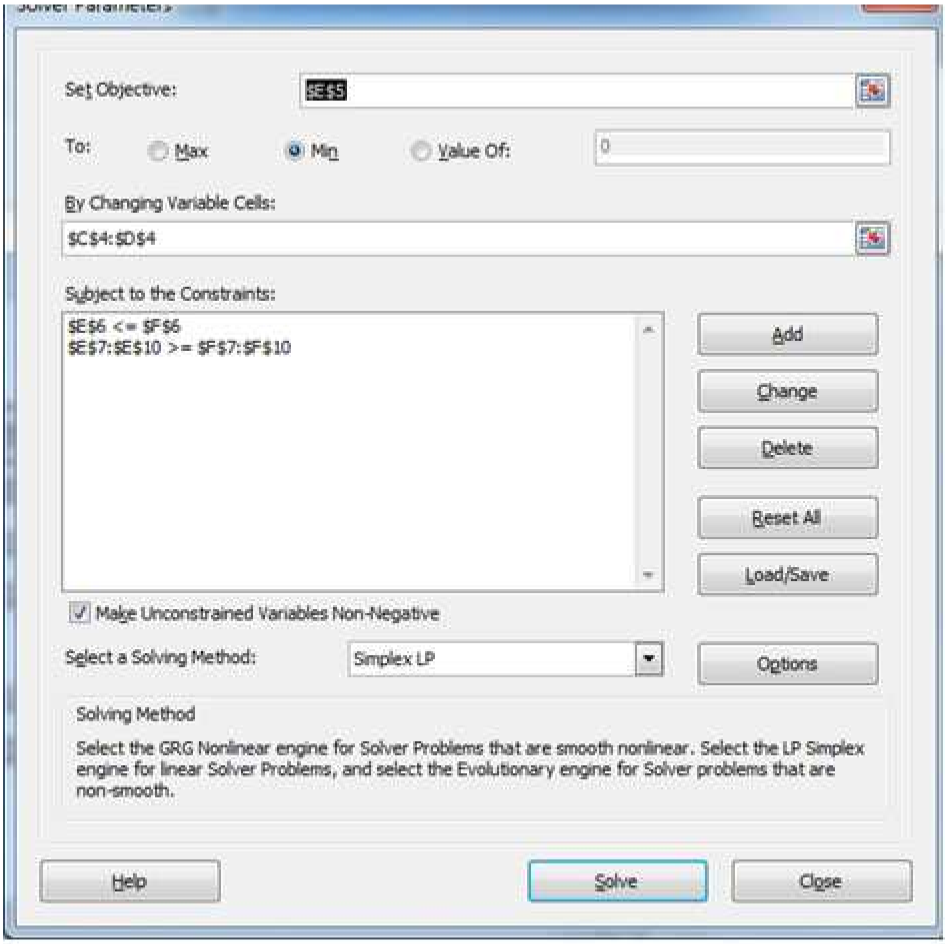
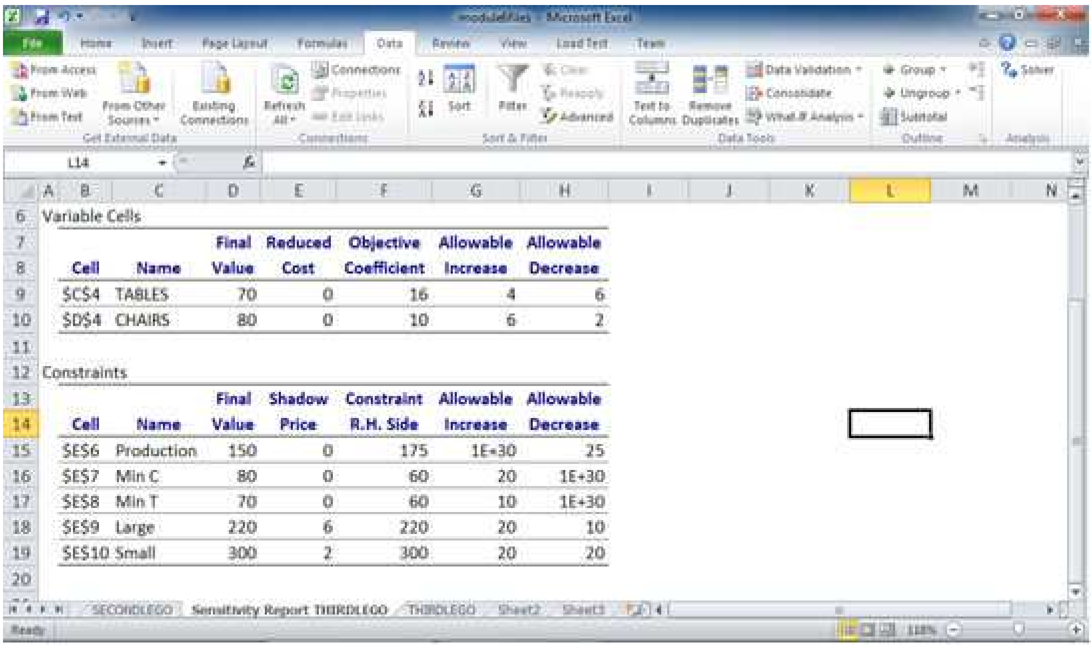


Figure 6.16



Variable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
C4	TABLES	70	0	16	4	6
D4	CHAIRS	80	0	10	6	2

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
E6	Production	150	0	175	1E+30	25
E7	Min C	80	0	60	20	1E+30
E8	Min T	70	0	60	10	1E+30
E9	Large	220	6	220	20	10
E10	Small	300	2	300	20	20

Figure 6.17

The SA report (Figure 6.17) is included to show it is relevant for all types of models (MIN or MAX) with all three kinds of constraints. Module 7 will detail how this information can help the modeler learn additional problem insight.

Reading Material: 6.6 – Summary

We saw the EXCEL Solver in action. The only way we can get comfortable with using EXCEL to solve our LP models is to practice. Please take a look at the assigned problems below and get all of the "kinks" out of using EXCEL. This will allow us to master the LP solution platform, making it second nature, and so we can turn our focus on the larger scale important business situations in which LP and other models can help make a difference.