Module 9: Forcing Variables to Have Integer Requirements

Reading Material: 9.1 – Why So Long?

It is okay if you have been a little bothered by our purposeful delay of discussing integer requirements for decision variables. I appreciate the interest in wanting to make the LP models totally realistic and applicable.

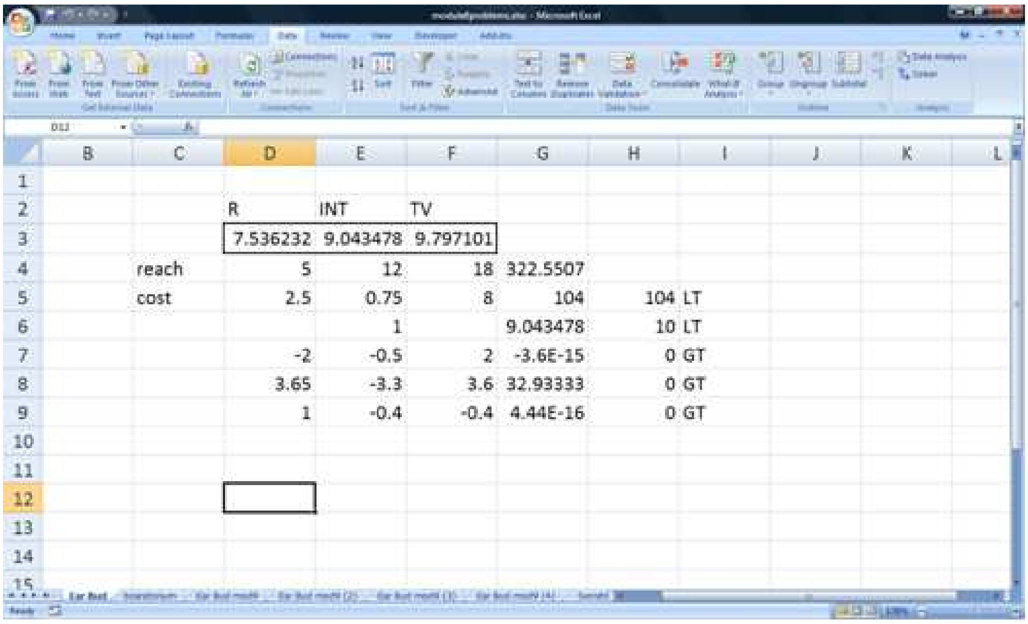
It is quite easy to add constraints forcing decision variables to be integers. When this additional requirement is added to models, the dynamics of the problem solution algorithm change dramatically. Traditional Sensitivity Analysis (SA) “goes away.” SA is an important component of our modeling foundation; thus, we completed it first before discussing integer requirements. With that complete, I believe Simba, “It is time!”

This is a relatively short module. First, a preliminary review using the Ear Bud example from the previous module illustrates that rounding and truncating fractional solution values of decision variables is inadequate to find optimal integer solutions. Then, the Solver constraints needed to force integrality are shown, along with a high-level discussion on the solution process employed by the Solver. This discussion helps provide insight into the optimal integer solutions that are found. The module concludes with a few more practice problems.

Reading Material: 9.2 – Rounding and Truncating Solutions – Infeasible and Dominated (Suboptimal) Solutions

9.2.1 Example Revisited – Ear Bud Advertising Campaign

In Module 8, an LP model was created and solved to find the optimal number of advertisements (satellite radio, internet, and TV) to maximize reach of the campaign for aural health. There were five constraints in the model related to cost, maximum number of internet ads, a women/men constraint, an old/young people constraint, and a composition constraint (40% radio ads). Figure 9.1 shows the optimal solution previously discussed without integer requirements. The optimal solution reached 322,550.7 people (Cell G4) by purchasing 7.536232 satellite radio ads, 9.043478 internet ads, and 9.79101 TV ads. Obviously, that is a non-implementable optimal solution.



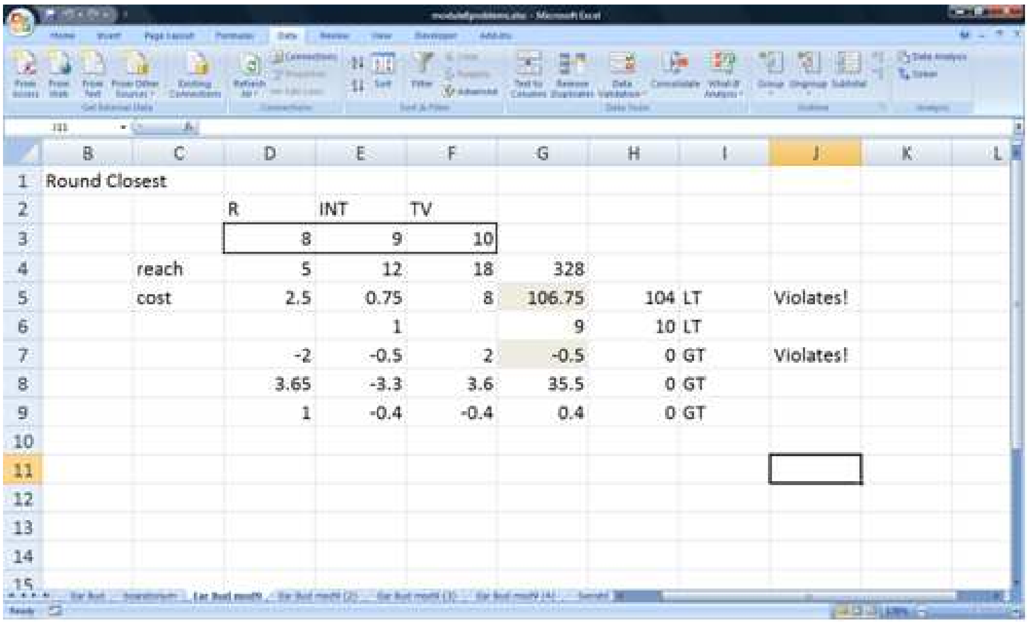
	B	C	D	E	F	G	H	I
1								
2			R	INT	TV			
3			7.536232	9.043478	9.797101			
4		reach	5	12	18	322.5507		
5		cost	2.5	0.75	8	104	104	LT
6				1		9.043478	10	LT
7			-2	-0.5	2	-3.6E-15	0	GT
8			3.65	-3.3	3.6	32.93333	0	GT
9			1	-0.4	-0.4	4.44E-16	0	GT

Figure 9.1

A reasonable strategy might seem to round solution values to the nearest whole number or to truncate the solution values (just throw away the fractional part of the solution value). Unfortunately, this does not work.

9.2.2 Using Our "Good" Judgment – Rounding to Nearest Integer = Infeasible

Figure 9.2 shows new "rounded" solution values for the model. RAD (satellite radio) would round up to 8 ads, internet ads (INT) would round to 9 ads, and the TV decision variable would round to 10 ads. By rounding, though, two of the constraints are violated (they are shaded). The budget constraint of 104 is exceeded, and the third constraint (women/men) is also violated. Thus, this potential solution is infeasible. Strike 1.

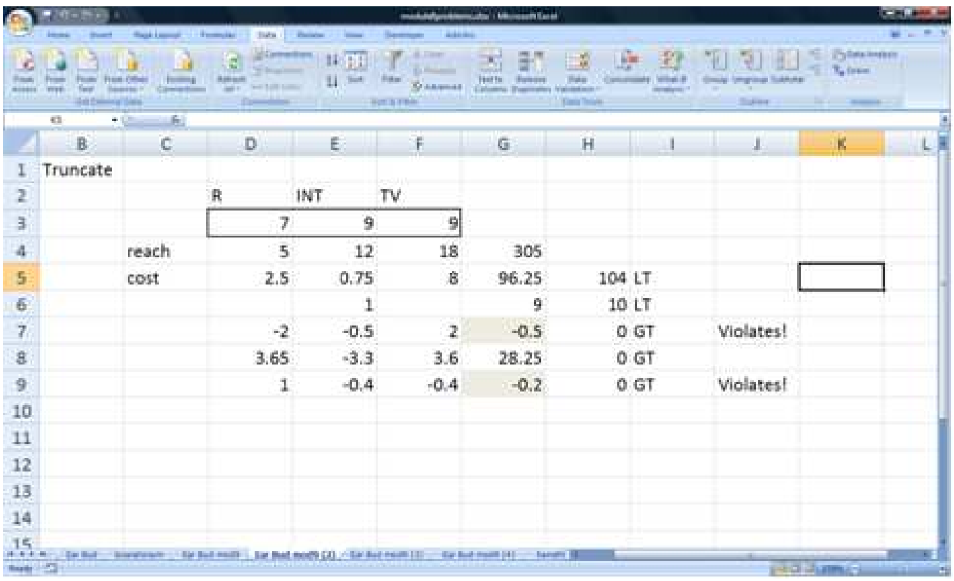


	B	C	D	E	F	G	H	I	J
1	Round Closest								
2			R	INT	TV				
3			8	9	10				
4		reach	5	12	18	328			
5		cost	2.5	0.75	8	106.75	104	LT	Violates!
6				1		9	10	LT	
7			-2	-0.5	2	-0.5	0	GT	Violates!
8			3.65	-3.3	3.6	35.5	0	GT	
9			1	-0.4	-0.4	0.4	0	GT	

Figure 9.2

9.2.3 Using Our "Good" Judgment II – Truncating Solution = Infeasible

Figure 9.3 shows new solution values if fractional parts of the solution values are truncated. RAD would then be equal to 7, INT would be equal to 9, and TV also equal to 9. Unfortunately, this solution is also infeasible, violating the third and fifth constraints (both demographic composition constraints). Strike 2.

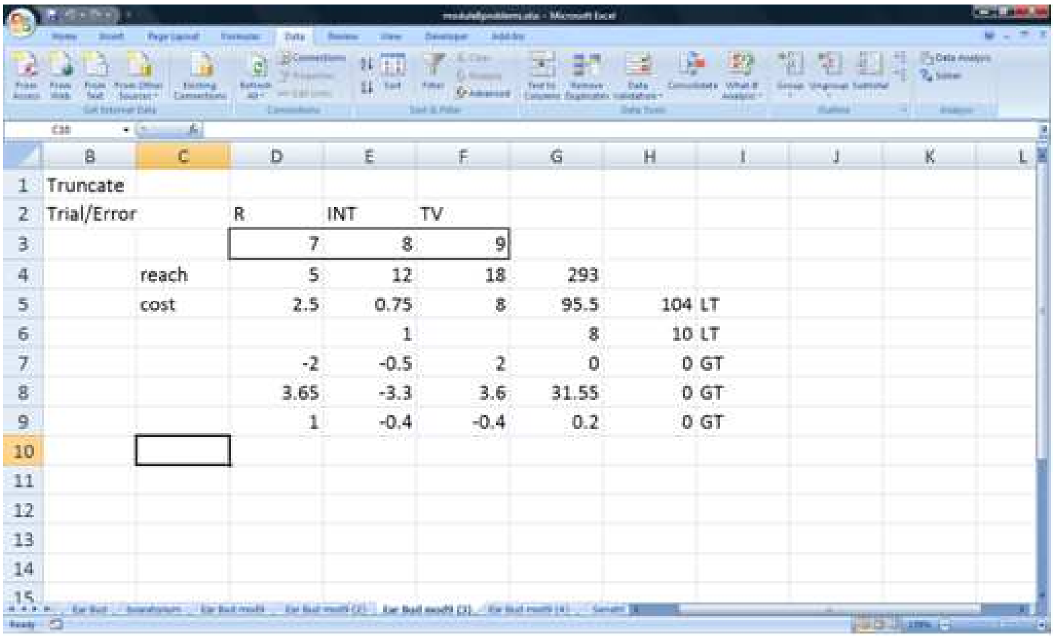


	B	C	D	E	F	G	H	I	J
1	Truncate								
2			R	INT	TV				
3			7	9	9				
4		reach	5	12	18	305			
5		cost	2.5	0.75	8	96.25	104	LT	
6				1		9	10	LT	
7			-2	-0.5	2	-0.5	0	GT	Violates!
8			3.65	-3.3	3.6	28.25	0	GT	
9			1	-0.4	-0.4	-0.2	0	GT	Violates!

Figure 9.3

9.2.4 Using Our "Good" Judgment III – Feasible, Suboptimal Solution

An observant modeler sees that by reducing the number of internet ads (INT) in the previous solution by 1, she can satisfy the third and fifth constraints and perhaps find the optimal integer solution. Well, the solution of RAD = 7, INT = 8, and TV = 9 is feasible, and provides reach of 293(000). Figure 9.4 shows this feasible solution.



	B	C	D	E	F	G	H	I
1	Truncate							
2	Trial/Error		R	INT	TV			
3			7	8	9			
4		reach	5	12	18	293		
5		cost	2.5	0.75	8	95.5	104	LT
6				1		8	10	LT
7			-2	-0.5	2	0	0	GT
8			3.65	-3.3	3.6	31.55	0	GT
9			1	-0.4	-0.4	0.2	0	GT

Figure 9.4

The only problem is that there is a better integer solution. We can applaud the insight the modeler used to find a feasible solution, but . . . Strike 3.

9.2.5 Let the Solver Solve

Figure 9.5 shows the best integer solution for the problem. The optimal mix of ads (RAD = 7, INT = 7, TV = 10) reaches 299(000) people and satisfies all constraints. It is a better solution than that found through trial and error. Although this may not seem like a big improvement, in a larger problem, the gap between a guess and the optimal integer solution might be substantial.

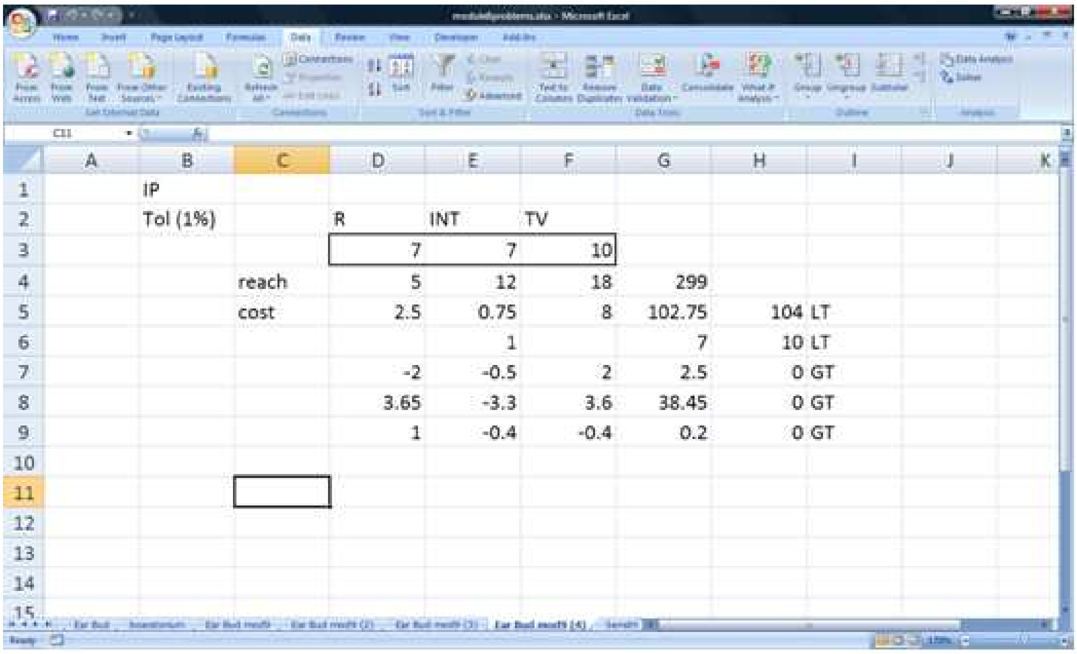


Figure 9.5

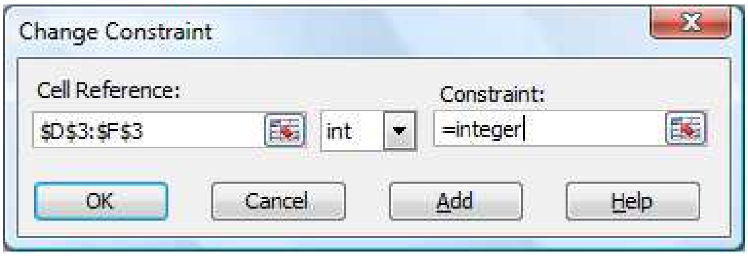


Figure 9.6

Figure 9.6 shows how to add integer requirements in the Solver. To do this, a constraint is added. The decision variable cells to be whole numbers are highlighted in the left-hand side (LHS) of the constraint dialog box. The constraint type selected is "INT." (There are no RHS inputs for these types of constraints – the word "=integers" will automatically be displayed.). This simple constraint will force the variables to be whole numbers.

You may notice that there is another menu choice for constraints – "BIN." That choice works the same as "INT" – it forces decision variables to be binary – 0/1 – a special case of integers. We will have a separate module on applications that use 0/1 variables later.

Reading Material: 9.3 – Miscellaneous Information about Integer Requirements

9.3.1 Why not right from the start?

Building fundamental understanding about the core concepts of LP (binding constraints, shadow prices, etc.) is very important. We can do that best in two dimensions or in small, uncomplicated models. SA and finding graphical solutions was relatively easy by allowing fractional solutions to our decision variables.

Look again at Figure 9.6, the best integer solution for the Ear Bud Advertising Campaign. Which constraints are binding? Where does the LHS = RHS?

Loaded question, wasn't it? It appears that none of the constraints are binding. How can that be?

From a two-dimensional graphing standpoint, if decision variables were forced to be whole numbers, there would not be a feasible region per se, just a number of feasible points (at the intersections of lines representing the whole numbers). Binding and non-binding constraint concepts are not readily apparent in these circumstances.

SA requires a continuous feasible region. Because that goes away with integer requirements, there is no SA report when variables are forced to be integers. The "simple algebra" that "falls out of the model" goes away. Again, if solution variables NEED to have integer values, they need to be integers! But you can see if in the text we immediately started forcing variables to be integers, we could not have properly covered some key insightful fundamentals.

SA that can be done in integer programming is simply good, old-fashioned brute force "what-if" analysis. For instance, you might see the impact of a number of $5 increases/decreases in budget for our advertising problem and gauge the approximate impact on the people reached. This requires the leg work of the modeler to run and rerun the model to assess the marginal impact of resources.

9.3.2 Solutions in Integer Programming

The solution algorithm employed by the EXCEL Solver (in Simplex mode) to get integer solutions is a "branch and bound" approach, another brute-force algorithm. It repeatedly solves modified LP models in a systematic function, gradually eliminating parts of the feasible region that contain fractional values. It can be very computationally intensive and has some parameter settings that need mention even for us as more novice users.

When in integer mode, the Solver notes which variables have fractional values (e.g., X4 = 3.4). For each decision variable with a fractional value, it creates two new sub-problems (LP models), each with an additional constraint (one with X4 <= 3, the other with X4 >= 4) and re-solves them as LP models. Thus, the Solver gradually slices and dices the feasible region until it finds the best integer solution (more on that later).

For small problems, the computational requirements are not noticeable. For larger problems in EXCEL – you can cook dinner while waiting for a model to solve (a little exaggeration – but not entirely).

In EXCEL, a tolerance factor (in EXCEL 2007) or a Integer Optimality % (EXCEL 2010) is used as a stopping rule. That is, when should the branch and bound process stop looking for better integer solutions than what has previously been found (recall, the slicing and dicing of the feasible region is going on for ALL variables that are fractional). As I personally have made the transition from EXCEL 2007 to EXCEL 2010, I have found interesting behaviors in the Solver. My understanding of the tolerance factor/Integer Optimality % is that the solution process will stop if the Solver finds an integer solution that is within that % of the optimal LP model (with fractions). Normally, for our sake, that is good enough, but it there is still a risk that you will get a nearly optimal solution instead of *the* optimal solution.

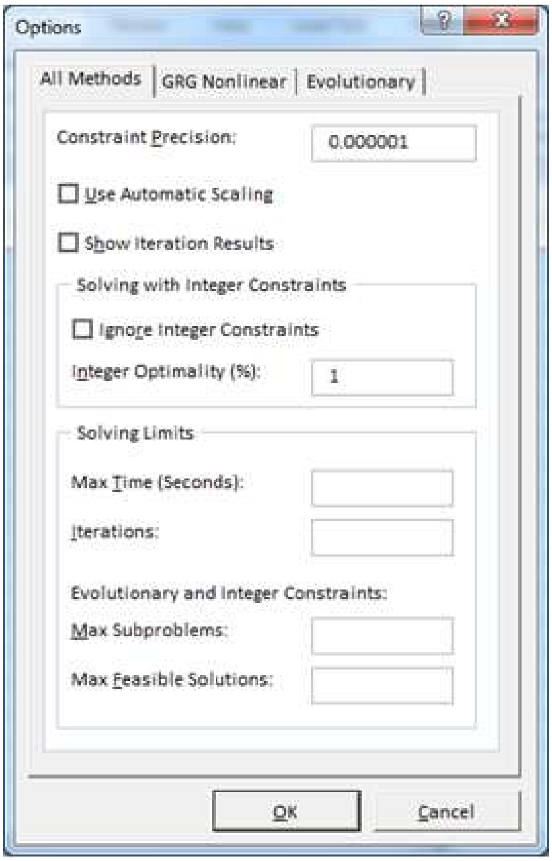


Figure 9.7

As of this printing, it appeared that EXCEL 2007 defaulted to 5% for this parameter and EXCEL 2010 defaults to 1%. Why wouldn't we specify 0%? Well, the IP solution is always going to be worse than the LP solution. For larger problems, the search can seem to go on indefinitely, and for basic solvers like what we have in EXCEL, sometimes we're going to want a solution that is done before dinner.

Messing around with algorithm parameters is a little beyond what I'd like to typically spend our time on. For now, I'd like to keep the default values, but I too am still learning how new versions of the Solver work. I might ask you to experiment just a little in the practice problems.

There are more sophisticated solvers (CPLEX, etc.) that provide more flexibility for the modeler in parameter settings and even algorithms. This is beyond the scope of this book.

Figure 9.7 is the options menu previously seen where the Optimality % is found (and other parameters). If you are using EXCEL 2010, always investigate the box entitled “Ignore Integer Constraints.” I have found backward compatibility issues. It seems when you take an EXCEL 2007 file and use the 2010 Solver (with integer requirements), this box shows up checked – and does not provide integer solution values! As of press time, I still am not 100% of the circumstances that cause this – so save yourself some angst and make sure you investigate the options screen when adding integer requirements.

There are other parameters in Figure 9.7 that relate to stopping rules for the solution algorithm – Max Subproblems, and Max Feasible Solutions. The other inputs shown under solving limits can be tweaked as well. For now, we shall just let them be.

Reading Material: 9.4 – Summary

If you want it, here it is (INT), come and use it! Just keep in mind what goes away (SA) and what you get with it (possibility of long solution times).

As always, practice problems follow next. Next module – more useful applications.