Linear Programming I: Maximization

Assignment 10 is on the last page.

Learning objectives:

1. Recognize problems that linear programming can handle.

2. Know the elements of a linear programming problem -- what you need to calculate a solution.

3. Understand the principles that the computer uses to solve a linear programming problem.

4. Understand, based on those principles:
   a. Why some problems have no feasible solution.
   b. Why non-linearity requires much fancier technique.

5. Be able to solve small linear programming problems yourself.

Linear programming intro

Linear programming is constrained optimization, where the constraints and the objective function are all linear. It is called "programming" because the goal of the calculations help you choose a "program" of action.

Classic applications:

1. Manufacturing -- product choice
   Several alternative outputs with different input requirements
   Scarce inputs
   Maximize profit

2. Agriculture -- feed choice
   Several possible feed ingredients with different nutritional content
   Nutritional requirements
   Minimize costs

3. "The Transportation Problem"
   Several depots with various amounts of inventory
   Several customers to whom shipments must be made
   Minimize cost of serving customers

4. Scheduling
   Many possible personnel shifts
   Staffing requirements at various times
   Restrictions on shift timing and length
   Minimize cost of meeting staffing requirements

5. Finance
   Several types of financial instruments available
   Cash flow requirements over time
   Minimize cost

Start with
The Manufacturing Problem -- Example 1

A manufacturer makes wooden desks (X) and tables (Y). Each desk requires 2.5 hours to assemble, 3 hours for buffing, and 1 hour to crate. Each table requires 1 hour to assemble, 3 hours to buff, and 2 hours to crate. The firm can do only up to 20 hours of assembling, 30 hours of buffing, and 16 hours of crating per week. Profit is \$3 per desk and \$4 per table. Maximize the profit.

The linear programming model, for a manufacturing problem, involves:

Processes or activities that can be done in different amounts

Constraints – resource limits

The constraints describe the production process – how much output you get for any given amounts of the inputs.

The constraints say that you cannot use more of each resource than you have of that resource.

Linear constraints means no diminishing or increasing returns. Adding more input gives the same effect on output regardless of how much you are already making.

Non-negativity constraints – the process levels cannot be less than 0.
This means that you cannot turn your products back into resources.

Objective function -- to be maximized or minimized

A linear objective function means that you can sell all you want of your outputs without affecting the price. You have elastic demand, in economics jargon.

Translating the words of Example 1 into equations (which is not a trivial task), we have this:

The objective function is Profit = $3x + 4y$
x is the number of desks   y is the number of tables

Constraints:

assembling	$2.5x + y \le 20$	The total assembling time, for example, is the time spent assembling desks plus the time spent assembling tables. That's 2.5x + 1y. This must be less or equal to the 20 hours available.
buffing	$3x + 3y \le 30$	
crating	$x + 2y \le 16$	
non-negativity	$x \ge 0$	
	$y \ge 0$	

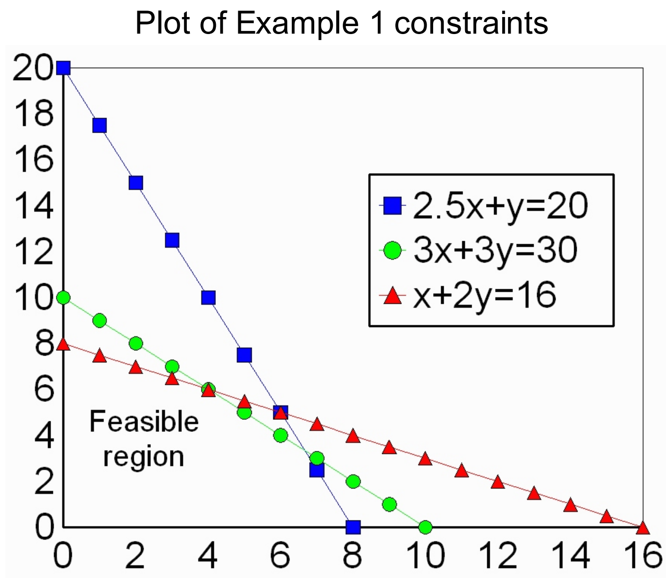
Graphical method of solution – for maximization

One way to solve a linear programming problem is to use a graph. The graph method lets you see what is going on, but its accuracy depends on how careful a draftsman you are.

1. Plot the constraints. Express each constraint as an equation. (Change the ≤ or ≥ to an =.)

2. Find the feasible region. It's below all lines for constraints that are ≤ and above all lines for constraints that are ≥.

Here, the feasible region is below the 2.5x+y=20, 3x+3y=30, and x+2y=16 lines – those are the ≤ constraints. It's above the y=0 line (the x-axis), and to the right of the x=0 line (the y-axis) – those are the ≥ constraints.

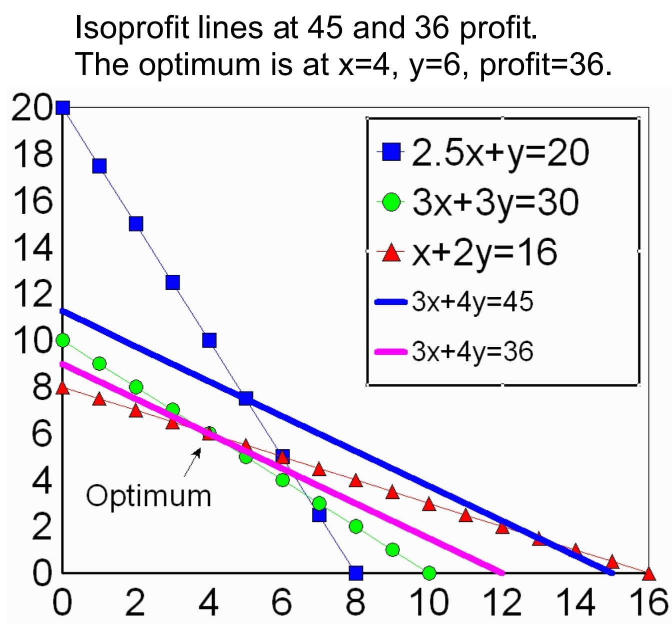


3. Superimpose **isoprofit** lines.

To get an isoprofit line, set the objective function equal to some arbitrary number. That gives you a linear equation in X and Y that you can plot on your graph. "Iso" means "the same." All the points on an isoprofit line give the same profit.

As a first try, in the diagram to the right, I drew an isoprofit line for a profit of 45. The equation for the line is 3x+4y=45. That is because we make $3 for each desk and $4 for each table. 3x+4y=45 is a line containing all the points that have a profit of exactly 45.

That 45 isoprofit line happened to be higher than the feasible region in the lower left corner, bounded by the constraints.



4. Find the highest value isoprofit line that touches the feasible region. Imagine moving that 3x+4y=45 line, parallel to itself, down and to the left. Move it down a little bit and you might have the line 3x+4y=44. Move it some more and you might have the line 3x+4y=43. Keep going until your line just touches a corner of the feasible area. Stop there, and you have the line 3x+4y=36, which is shown in the diagram.

All isoprofit lines have the same slope. They differ only in how high they are. The slope of an isoprofit line depends on the ratio of the x good's profit-per-unit to y good's profit-per-unit.

The profit amount for the isoprofit line that just touches the feasible area is the most profit you can make. The X and Y coordinates of the point where the isoprofit line touches tells you how much of x and y to make.

Why do you stop with the the isoprofit line that just touches the feasible area? Higher isoprofit lines have no feasible points on them. We cannot use those. Lower isoprofit lines, that touch more of the feasible area, have less profit than the one that just touches a corner. We want the most profit that is feasible, so we want the line that just touches a corner..

The solution is x = 4, y = 6, and the profit is 36.

Enumeration method of solution

The enumeration method is another way of solving a linear programming problem. It gives an exact answer that does not depend on your drawing ability. However, it can involve a lot of calculating. To understand the enumeration method, we start with the graph method.

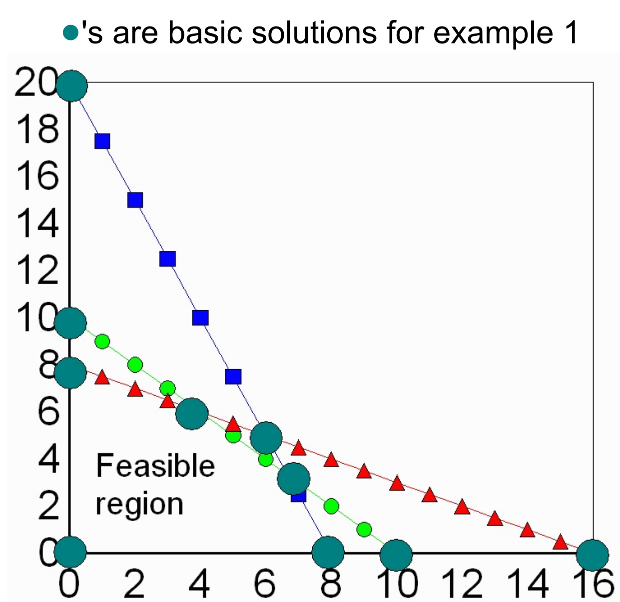
The feasible region in the diagram above is convex with straight edges. This is always true in linear programming problems. This implies the **extreme point theorem**: If a feasible region exists, the optimal point will be a corner of the feasible region. When the isoprofit line just touches the feasible region, it will be touching at a corner. (It is possible for the isoprofit line to touch a whole edge, if one of the constraint lines is parallel to the isoprofit lines. That whole edge will include two corners, so the general theorem about corners still applies.)

The corners of the feasible region are points where constraints intersect. If we can find all of the intersections of the constraints, we know that one of those intersections must be an optimal point.

Some jargon: Each point where the constraints intersect is called a **basic solution**.

A basic solution for a 2-product problem like ours is where any two of the constraint lines intersect.

(If our problem had three products, each basic solution would be where three of the constraint planes intersect. For a problem with **n** products, where **n** is more than 3, each basic solution would be where **n** of the constraint hyperplanes intersect.)
The non-negativity equations count as constraints, too, when identifying basic solutions. Example 1 has 5 constraints, and therefore 10 basic solution intersections. They are shown in the diagram.



Some of the basic solutions are corners of the feasible region. Other basic solutions are outside of the feasible area.

A **basic feasible solution** is a basic solution that satisfies all the constraints. In other words, a basic feasible solution is a corner of the feasible area. In Example 1 there are 5 basic feasible solutions, the five corners of the feasible region.

The extreme point theorem implies that one of the basic feasible solutions is the optimal point.

To get the solution to this linear programming problem, we just have to test all of the basic solutions to see which are feasible and which are not. Once we have the basic feasible solution, we calculate the profit for each one and pick the highest.

Summarizing the enumeration method (for **n** dimensions, representing **n** different products):

1. Make all possible groups of **n** equations from the constraints.
2. Solve each group of equations. This gives you all the basic solutions.
3. Test each solution against the other constraints.
4. Throw out any solutions that aren't feasible. The ones that are left are the basic feasible solutions.
5. Calculate the profit for each basic feasible solution.
6. Pick the highest profit point as your answer.

This can be done entirely with algebra. You don't need to draw a diagram.

Enumeration solution to Example 1:

n=2, so we solve the equations in pairs. There are 5 constraint equations, counting the non-negativity conditions, so there are 10 possible pairs of equations. That means 10 solutions.

Solutions x	y	Slack in the Constraints			Feasible Solution?	Profit	
0	0	20	30	16	Yes	0	
0	8	12	6	0	Yes	32	
0	10	10	0	-4	No		
0	20	0	-30	-24	No		
4	6	4	0	0	Yes	36	◂Winner!
6	5	0	-3	0	No		
6.667	3.333	0	0	2.667	Yes	33.33	
8	0	0	6	8	Yes	24	
10	0	-5	0	6	No		
16	0	-20	-18	0	No		

The table above lists all ten intersections, the ten basic solutions. The middle columns test to see if each one uses too many resources. Negative numbers under "Slack in the Constraints" indicate that the solution uses too much of a resource, and is therefore not feasible. For the solutions that are feasible, the profit is calculated. The winner is the row with the highest profit.

The advantages of the enumeration method over the graphics method are:

1. Enumeration can be used for problems with more than 2 dimensions.
2. It gives an exact answer.
3. You can program a computer to do it, because it is entirely algebraic.

The disadvantage of the enumeration method:

The math grows rapidly with number of equations. If you have **c** constraints and **p** processes, the number of intersections is **(c+p)! / c!p!** . For example, if you have three constraints and two variables, you have ten intersections. If you have six constraints and six variables, you have 924 intersections. And each intersection requires that much more calculation to find.

Simplex method -- for maximization

The simplex method is so named because the shape of the feasible region, a solid bounded by flat planes, is called a "simplex."

The simplex method is an algorithm. It speeds up the enumeration method by moving step-by-step from one basic feasible solution to another with higher profit until the best is found. The simplex method is presented here for your edification. You will not be asked to do the simplex method by hand like this. The idea of this presentation is to show you that the simplex method has a series of steps that are suitable for a computer to do.

The simplex method requires first putting the problem into a standard form. You change the constraints from inequalities into equalities by adding slack (or surplus) variables. You add one slack variable for each constraint. It represents how much of your capacity in that particular regard that you aren't using.

Example 1 constraints become

$2.5x + y + s_1 = 20$
$3x + 3y + s_2 = 30$
$x + 2y + s_3 = 16$

We want to express these equations in matrix form. To do this, we include all the variables in every equation, putting in 0's where needed, like this:

$$2.5x + 1y + 1s_1 + 0s_2 + 0s_3 = 20$$
$$3x + 3y + 0s_1 + 1s_2 + 0s_3 = 30$$
$$1x + 2y + 0s_1 + 0s_2 + 1s_3 = 16$$

This system of three equations in five unknowns has 10 solutions. Each one corresponds to one of the basic feasible solutions shown on pages 4 and 5. One of those 10 is relatively easy to spot. Suppose, in the above equations, x and y are both 0. Then the three equations reduce to $1 \times s_1 = 20$, $1 \times s_2 = 20$, and $1 \times s_3 = 16$. This is the solution corresponding to making no desks or tables and leaving all of the resources unused ("slack").

Arrange each equation so that the variables are in the same order in each one. You can then leave out the variable names and just keep the numbers.

2.5	1	1	0	0	20
3	3	0	1	0	30
1	2	0	0	1	16
-3	-4	0	0	0	0

← The objective function is added at the bottom. In that row, the numbers are the negatives of their actual values. You fill out the rest of the bottom row with 0's so that the last row has the same number of elements as the others. The 0's under the slack variable columns indicate that leaving slack gains nothing (nor is there any disposal problem.) The 0 in the lower right says that if we produce 0 we make 0 profit.

The simplex method solves these equations in groups of n (n=2 in this case), like the enumeration method. The simplex method improves on the enumeration method by using the bottom row and the right column to proceed systematically from one feasible basic solution to the next, always increasing the value of the objective function until a maximum is reached. This shortens the calculations required.

The simplex method is thus "iterative." You repeat it until it tells you to stop. Here's what you repeat.

1. Choose a "pivot column" -- the column that has the smallest (biggest negative) number in the last row. (The last column on the right is the totals. It can't be used.)

2. Choose the "pivot row" -- divide each element in the right-most column by the corresponding element in the pivot column. Choose the row with the smallest quotient that is bigger than 0. (Not the special bottom row.)

2.5	1	1	0	0	20	20/1 = 20	
3	3	0	1	0	30	30/3 = 10	
1	<u>2</u>	0	0	1	16	16/2 = 8	← Pivot row
-3	-4	0	0	0	0		
	↑						

The most negative element of the bottom row indicates the pivot column.

3. The intersection of the pivot row and pivot column is the pivot element.

Here, the pivot element is the 2 in the 3rd row, 2nd column.

4. Change the pivot element to 1 by dividing everything in its row by 2. That'll leave an equation that's still true. ($0.5x + 1y + 0s_1 + 0s_2 + 1s_3 = 8$)

1 2 0 0 1 16

becomes

0.5 1 0 0 0.5 8

5. Fill out a new matrix by using Gaussian elimination to make all the other elements in the pivot column 0. For each other row, multiply this new row by the element in the pivot column, then subtract across.

In this example, you make a new first row by subtracting the new pivot row from it, element by element.

Each element in the new second row is the old second row minus 3 times the corresponding element in the new pivot row. Do the same to the bottom row. Multiply the new pivot row by 4 and add each element. You get:

2	0	1	0	-0.5	12
1.5	0	0	1	-1.5	6
0.5	1	0	0	0.5	8
-1	0	0	0	2	32

This is still interpreted as a set of equations (the first three lines, that is). If you put back in the x's, y's, s's, +'s, and ='s, you get:

$$2x + 0y + 1s_1 + 0s_2 - 0.5s_3 = 12$$
$$1.5x + 0y + 0s_1 + 1s_2 - 1.5s_3 = 6$$
$$0.5x + 1y + 0s_1 + 0s_2 + 0.5s_3 = 8$$

This set is equivalent to the first set in that they have the same ten solutions that are in the table on page 5. What's different is that the first set made the solution $x = 0$, $y = 0$, $s_1 = 20$, $s_2 = 30$, $s_3 = 16$ obvious (relatively). This set makes a different solution obvious, namely $x = 0$, $y = 8$, $s_1 = 12$, $s_2 = 6$, $s_3 = 0$.

Here is the tableau again, with the bottom row back in, and labels for each column:

x	y	s_1	s_2	s_3	
2	0	1	0	-0.5	12
1.5	0	0	1	-1.5	6
0.5	1	0	0	0.5	8
-1	0	0	0	2	32

The way to read this tableau is to look for the columns that have only 1's and 0's. Those are the variables now included in the answer ("basis"). The corresponding number in the right column tells the amount to use. The lower right corner number tells how much money we'll make. This tableau says that we make 8 units of y and leave 12 slack in the first constraint and 6 slack in the second constraint, we'll make $32.

6. If there are no negative numbers left in the last row, stop. Otherwise, go back to step 1 and go through the steps again. Here we have a -1 in the bottom row so we're not done.

2	0	1	0	-0.5	12	12/2 = 6
<u>1.5</u>	0	0	1	-1.5	6	6/1.5 = 4 ← Smallest quotient.
0.5	1	0	0	0.5	8	8/0.5 = 16
-1	0	0	0	2	32	
↑						

The most negative number in the bottom row indicates next pivot column.

The next pivot column is the first column. Divide each number in the right-most column by the corresponding number in the pivot column. The smallest quotient is in the second row, so the next pivot row is the second row.

Divide everything in row 2, the pivot row, by 1.5, the pivot element, to change it to:.

1	0	0	.667	-1	4

Subtract 2 times this from the first row. Subtract 0.5 times this from the third row. Add 1 times this to the last row. The transformed matrix is:

0	0	1	-1.3	1.5	4
1	0	0	.67	-1	4
0	1	0	-.33	1	6
0	0	0	.67	1	36

When there are no negative numbers left in the last row, stop. We can stop now.

To decipher this, put column headings on.

x	y	s_1	s_2	s_3	=
0	0	1	-1.3	1.5	4
1	0	0	.67	-1	4
0	1	0	-.33	1	6
0	0	0	.67	1	36

We get our amounts to use from the columns with all 1's and 0's.
In the first row, the 1 is under s_1. The 4 on the right means that we use 4 of slack variable 1 (assembling). In other words, in our optimal solution we'll have 4 hours of assembly time left over.

The second row has the 1 under x. The 4 on the right means that we produce 4 of x.

The third row says we produce 6 of y.

The bottom right element tells us we'll make $36.

Shadow prices – what the resources are worth to you

The elements at the bottom of the other slack columns tell us how much money an extra unit of those resources would be worth to us. The 1 at the bottom of the s_3 column tells us that if we had one more of the third resource we could make $1 more profit. The other entries in the s_3 column tell us how we'd change our production program if we had 1 more of the third resource (crating). The first row has a 1 s_1 and a 1.5 under s_3. This means that we would increase our slack in resource 1 by 1.5. The second row has a 1 under x and a -1 under s_3. This means that we would make one less x (desk). The third row has a 1 under y and a 1 under s_3. This means that we would make 1 more y.

Similarly, the 0.67 at the bottom of the s_2 column says that if we had one more unit of the second resource (buffing), we could rearrange our production plan to make $0.67 more.

These amounts-that-we'd-be-willing-to-pay are called **shadow prices**. We want to buy more of any resource whose actual market price is less than our shadow price. For example, if we could buy another hour of buffing time for less than $0.67, we should do so.

That is the simplex method by hand. Obscure, but systematic, and less work than the enumeration method. And it gives us the shadow prices as a bonus!

Now we get to how you will actually solve the problem, which is to use a computer. Specifically, you will use the simplex method optimization that is available in Excel and other full-featured spreadsheets.

The Excel solver, for linear maximization

Excel lets you set up your problem in a spreadsheet. You then tell Excel which cells have the constraints and the objective function. On command, Excel executes the simplex method and displays the solution. (Except for Excel's bugs. We will get to those later.)

Your first step should be to lay out all your equations on paper – your objective function and your constraints. Here's Example #1 again:

The objective function is Profit = 3x + 4y
x = number of desks   y = number of tables

Constraints:

assembling	$2.5x + y \leq 20$
buffing	$3x + 3y \leq 30$
crating	$x + 2y \leq 16$
non-negativity	$x \geq 0$
	$y \geq 0$

Start Excel. Start a new spreadsheet and enter the information like this:

┘	A	B	C	D	E
1		Desks	Tables	Total	
2	Amount to make	0	0		
3	Unit profit	3	4	=B3*$B$2+C3*$C$2	
4					
5	Constraints				Limit
6	assembling	2.5	1	=B6*$B$2+C6*$C$2	20
7	buffing	3	3	=B7*$B$2+C7*$C$2	30
8	crating	1	2	=B8*$B$2+C8*$C$2	16

The formulas in column D should display 0's, like this:

┘	A	B	C	D	E
1		Desks	Tables	Total	
2	Amount to make	0	0		
3	Unit profit	3	4	0	
4					
5	Constraints				Limit
6	assembling	2.5	1	0	20
7	buffing	3	3	0	30
8	crating	1	2	0	16

(An alternative way to write the D column formulas is to use sumproduct. For example, D3 could be `=SUMPRODUCT(B3:C3,$B$2:$C$2)`.)

This layout has column for each type of item. B is for desks. C is for tables. The 0's in B2 and C2 are where the amounts to make will go.

B3 and C3 have the profit-per-unit of desks and tables, respectively. D3 has the formula for total profit, which is what we want to maximize. Right now, it calculates to 0. The $ signs let you copy D3 and paste to D6:D8.

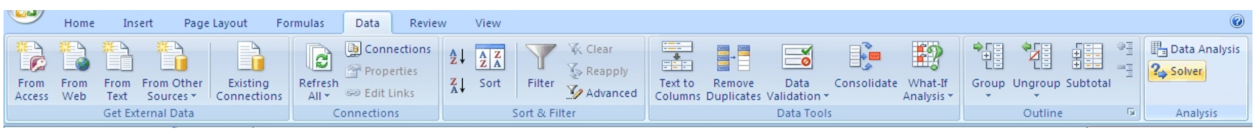
In rows 6-8, columns B and C show how much of each resource – type of labor – that one desk and one table use. The D column shows the total of each resource used up. The limit on how much you can use of each resource goes in column E.

This layout is very much like the What If example. We could go ahead and try different numbers in B2 and C2 and see what they do to profit and resource use. Instead, let's get Excel to do the work.

In Excel 2003 and older, click Tools on the top menu, then click Solver...

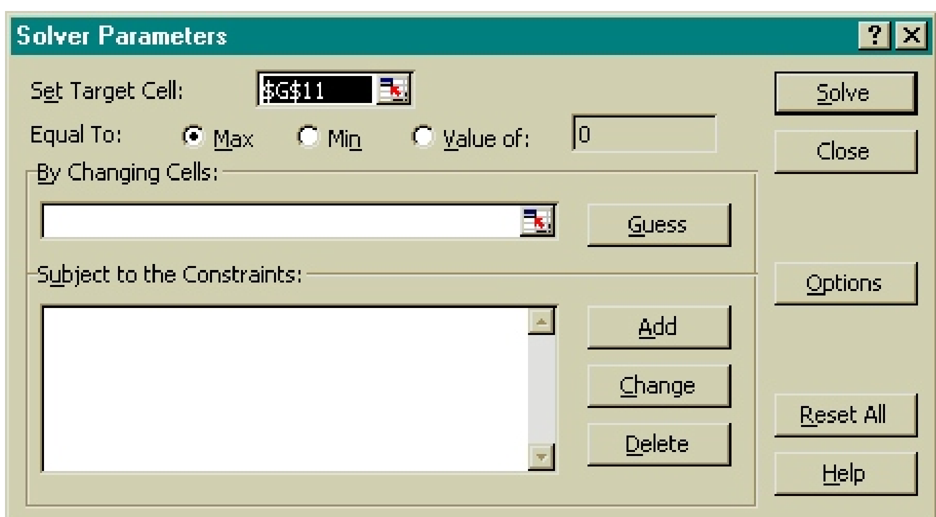
> If Solver... isn't on the Tools drop-down menu, you must install it. Click on Add ins... (lower down on the Tools menu). Put a check in the Solver box. If there is no Solver box, you'll need your original CD. Run the setup program and tell it to add Solver. Solver is one of the options under Excel. (If your copy of Excel was installed over a network, ask your network guru.)

In Excel 2007, click the Data tab and then look for Solver at the far right. Click on it.



If Solver is not there, please see the syllabus for how to install it. (Mac Office 2008 does not have Solver.)

This dialog box will appear.



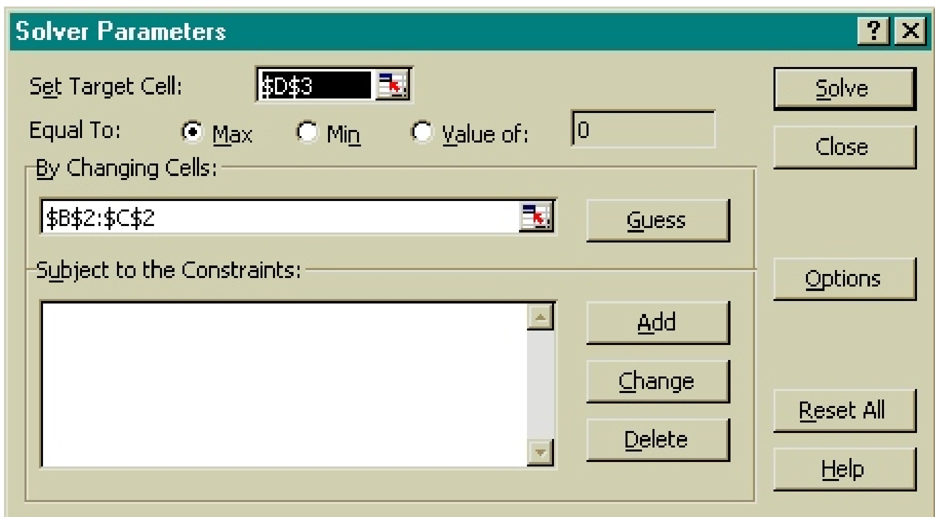
First, do the Set Target Cell. This cell is for the quantity that you're trying to maximize or minimize. When you start Solver, Excel fills that space in with a reference to the cell your selector happened to be on.

To set the Target Cell, click on the on that line. The dialog box will hide. We want to maximize what's in D3, so click on D3. (If the Solver Parameters bar is in the way, click and drag it away with your mouse.) Press Enter. $D$3 will appear in the Set Target Cell box. (You could have just typed `D3` in the Set Target Cell box, but using the pointer is more fun.)

Next, verify that the Equal To: line has Max checked. If not, check Max.

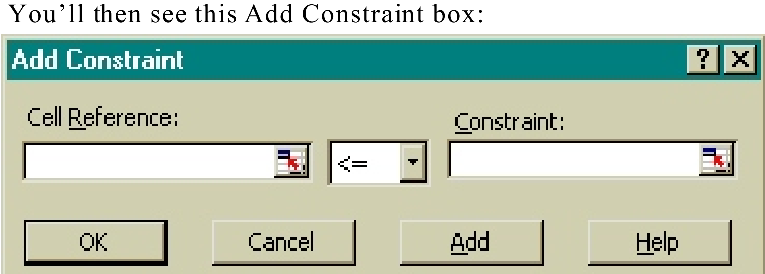
Click on the near the By Changing Cells: box. Here is where you tell Excel which cells it can change to optimize what's in the Target Cell. In this example, the variable cells are the numbers of desks and tables made. When the dialog box hides, move the mouse pointer to B2. Press and hold down the left mouse button. Move the pointer to C2, dragging the black area as you go. (Another way to highlight B2:C2 is to click on B2, press and hold Shift and click on C2.) Release the mouse button, press Enter , and B2:C2 will appear in the By Changing Cells: box.

Here's where we are, so far.



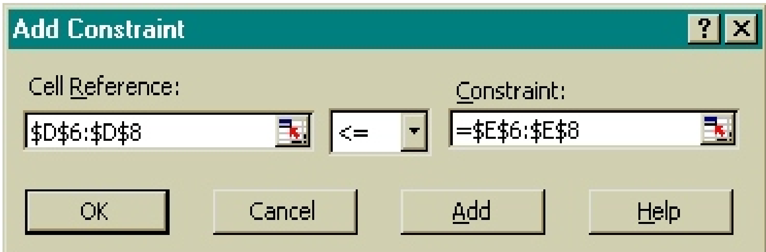
Next come the constraints. Click on the Add button.

You'll then see this Add Constraint box:



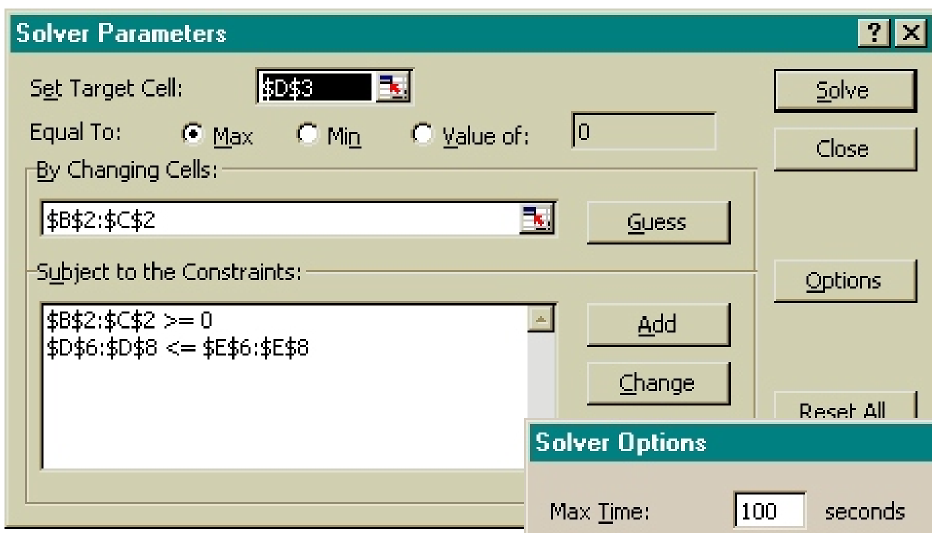
Click on the for Cell Reference: This Add Constraint box hides.

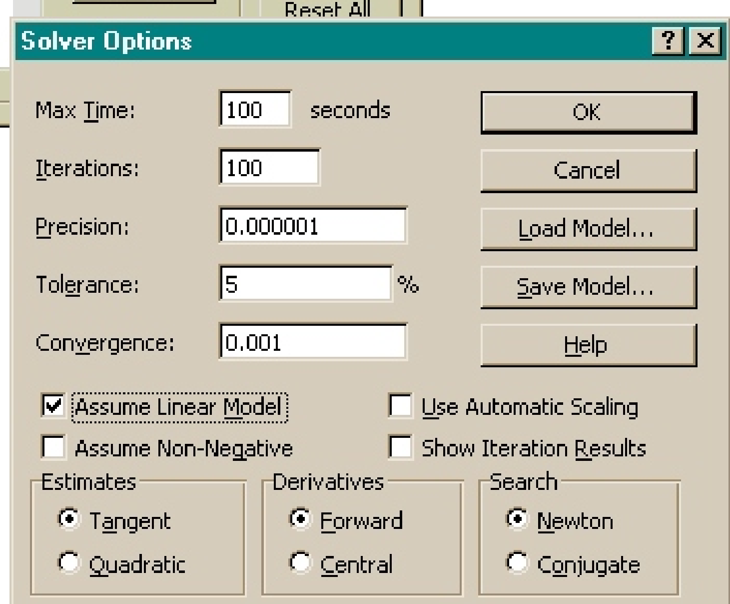
We can add the constraints one-at-a-time, or we can save a little time by adding all three resource constraints at once. Click on D6, which has the formula for the total assembling hours used. Hold the left mouse button down and drag to D8, highlighting the three cells D6:D8. Press Enter , and the Add Constraint dialog box reappears, with D6:D8 in the Cell Reference box. Put E6:E8 in the Constraint box, by the same technique. The Add Constraint box should look this:



Click on the Add button in this box. We now add the non-negativity constraints, the ones that say that all of the amounts to make must be zero or higher. Click on the for Cell Reference. Then click on B2 and drag across to highlight B2:C2. Press Enter , and B2:C2 should be in the Cell box as shown. Don't press OK yet! Instead, click on the <= operator in the middle. Change it to >=, because we want B2 and C2 to be bigger than or equal to 0. Next, click on the Constraint: white box (*not* the symbol next to it). Type 0 (zero) in that box. Press Enter or click on the OK button. That's all the constraints!

Here's what the Solver box looks like with all the information filled in.





One more thing we should do before we Solve: Click on the Options button. This brings up the Solver Options dialog box. Click to place a check mark next to Assume Linear Model .

Click on OK.

Back in the Solver Parameters box, click on Solve .

Excel will pause for a second or so, then show the answer on your screen. The answer may be obscured by the Solver Results dialog box (see below for what that box looks like). If that box is blocking your view, do not close the box. Instead, click on the box’s title bar (the colored band at the top that says Solver Results) and drag with your mouse to move the box enough so that you can see the following:

˩	A	B	C	D	E
1		Desks	Tables	Total	0
2	Amount to make	4	6		
3	Unit profit	3	4	36	
4					
5	Constraints				Limit
6	assembling	2.5	1	16	20
7	buffing	3	3	30	30
8	crating	1	2	16	16

Tah Dah! The amounts to make are filled in. The maximum possible profit is 36. The variable cells show how many desks and tables to make to reach that profit. You can see how many resources are used, and that they do satisfy the constraints.

If the table doesn’t look like the above, click on the Restore Original Values button in the Solver Results box. Then check your spreadsheet formulas and the Solver Parameters for errors.

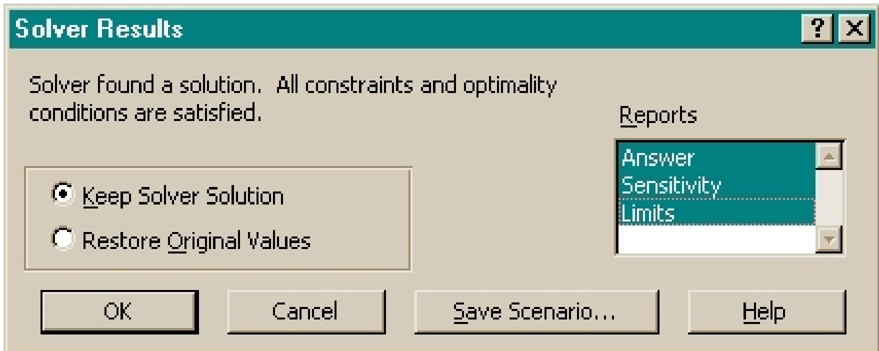
Error messages or weird results from Solver

Small numbers, like 1.4E-17, or numbers that seem slightly off, like 299.999997 where 300 should be, are due to round-off error in Excel’s algorithm. Try selecting Solve again so Excel can re-optimize. If that does not get rid of the tiny numbers or discrepancies, round them off yourself in your write-up.

Extremely large numbers, like 1.2E+308, mean that you made a mistake either in your spreadsheet or in the Solver dialog box. Inspect the Answer Report. Verify that the Target Cell, the Adjustable Cells, etc., are correct. Check that the constraints are correct and the <= or >= signs are pointing the right ways. Be sure that you're not maximizing the solution when you mean to be minimizing, or vice versa.

If the solution has no bound or no feasible solution exists:.

1. Verify that you're not minimizing the solution when you mean to be maximizing, or vice versa.
2. Be sure all the constraints are there and the <= and >= signs are pointing the right ways.
3. Put in non-negativity constraints or check the non-negative box in the Options (if you haven't done that already), to force all your solution cells to be ≥ 0.



If the spreadsheet looks OK, click in the Solver Results box on the words Answer, Senstivity, and Limits in the Reports section to select them, as shown here. Then click on OK.

Excel will then add three tabs to the bottom of the screen, indicating three new spreadsheet pages. There is one new spreadsheet page for each report you ordered.

If you get an error message when you try to get the reports

If you are using Excel 2007 with Windows Vista, an error message box may pop up when you try to get the Answer, Sensitivity, or Limits reports. There is nothing wrong with what you did. This is a bug in Excel (or Vista. Take your pick.)

Here is how to temporarily fix it: Remove Solver from Excel and then put it back in.

1. Click the Office Button. (It's circular, at the upper left corner of the Excel window.)
2. Click the Excel Options button. (Rectangular. Near the bottom of the pop-up window.)
3. Click on Add-Ins. (In the list on the left.)
4. See that next to Manage: it says Excel Add-ins. (If it doesn't say that, click on the little box and fix it.)
5. Click Go...
6. Uncheck Solver Add-in.
7. Click OK.

Solver should be gone from the menu.

8. Repeat the above, except that this time check Solver-Add-in.

Solver should be back on the menu.

You may have to do this every time you use Solver on a new file. Sorry!

Now Solve again and you should be able to get the reports.

Solver's Reports

Click on the Answer Report tab at the bottom of your spreadsheet, to see:

The Answer Report arranges neatly some results that we can see from the spreadsheet.

If your results do not make sense, look at the cell references and the constraints as listed in the Answer report to be sure that you have set up the problem correctly.

Target Cell (Max)

Cell	Name	Original Value	Final Value
$D$3	Unit profit Total	0	36

Adjustable Cells

Cell	Name	Original Value	Final Value
$B$2	Amount to make Desks	0	4
$C$2	Amount to make Tables	0	6

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$D$6	assembling Total	16	$D$6<=$E$6	Not Binding	4
$D$7	buffing Total	30	$D$7<=$E$7	Binding	0
$D$8	crating Total	16	$D$8<=$E$8	Binding	0
$B$2	Amount to make Desks	4	$B$2>=0	Not Binding	4
$C$2	Amount to make Tables	6	$C$2>=0	Not Binding	6

Next, click on the Sensitivity Report tab at the bottom of your spreadsheet. The Sensitivity Report should look like this:

Adjustable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
$B$2	Amount to make Desks	4	0	3	1	1
$C$2	Amount to make Tables	6	0	4	2	1

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
$D$6	assembling Total	16	0	20	1E+30	4
$D$7	buffing Total	30	0.666666667	30	3	6
$D$8	crating Total	16	1	16	4	2.666666667

The sensitivity report has the **Shadow Prices** (which may be called Dual Values or Lagrange multipliers, depending on your spreadsheet and the options you chose). There is a shadow price (or dual value or Lagrange multiplier) for each constraint.

Each constraint's shadow price (or dual value or Lagrange multiplier) tells you how much your objective function's total value change if you were to increase that constraint's limit by one. In this example, the objective is profit. The objective function's total value is your total profit. The shadow price for each constraint is therefore how much profit would go up if you had one more hour available of that constraint's type of labor. The shadow price also shows how much your profit would decrease if that constraint’s limit were one less.

One use of the shadow price is this: It tells you what it would be worth to you to get one more hour of any one type of labor. For example, the D8 crating constraint has a shadow price of $1. If you could get one more hour of crating time, you could change your production plan in a way that makes $1 more in profit. The most you would be willing to pay for another hour of crating time would therefore be $1.

The D7 constraint has a positive shadow price of two-thirds of a dollar. The buffing constraint is binding. You are using all that you have. Adding one more hour of buffing time would be worth about $0.67, because your profit would be that much higher.

The D6 constraint has a zero shadow price. This means that adding one more hour of assembly time does not affect profit. The assembling constraint is not binding. There is unused assembly time. You would not be willing to pay anything for more assembly hours, because you are not using all of what you already have.

You can confirm how the shadow prices relate to profit by going back to the spreadsheet and changing the E7 entry from 30 to 31. Solve again and you'll find that the maximum profit is 36.6667. Sure enough, one more hour of buffing time is worth $0.67 in increased profit. It is worth it to get another hour of buffing time only if it costs you less than $0.67. (Those of you who read through the simplex method section will notice that this is the same number we got for this from that method.)

The Limits Report is not particularly interesting, unless I’m missing something. Am I?

Assignment 10

1. You make decorative stones for landscaping. A ton of coarse stones requires 2 hours of crushing, 5 hours of sifting, and 8 hours of drying. A ton of fine stones requires 6 hours of crushing, 3 hours of sifting, and 2 hours of drying. The coarse stones sell for $400 per ton. The fine stones sell for $500 per ton. In a work week your plant is capable of 36 hours of crushing, 30 hours of sifting, and 40 hours of drying.

Use the graphical method first, to answer a. below. Then use your spreadsheet to verify your answer and get the numbers you need for b. and c.

> *For full credit for your graph, draw the constraints and at least one iso-revenue line. Draw a circle or an arrow to indicate the optimal point.*

Determine:

a. How much of each kind of stones you should make to maximize your revenue.
b. How much revenue you'll make at the maximum.
c. How much it would be worth to you to get another hour of crushing time, sifting time, or drying time.

2. You make three kinds of computers: Cheap, Good, and Deluxe. These sell for $1500, $2000, and $2400. The Cheap model requires 3 hours for circuit board installation and 1 hour to fit the peripheral equipment. The Good model requires 1 hour for circuit boards and 5 hours for peripherals. The deluxe model requires 3 hours for circuit boards and 2 hours for peripherals. You have 120 hours available for circuit board work and 60 hours for fitting peripherals. Determine:

a. How much of each kind of computers you should make to maximize your revenue.
b. How much revenue you'll make at the maximum.
c. How much it would be worth to you to get another hour of circuit board assembly time or peripheral fitting time.

This time you'll have to rely on your spreadsheet rather than the graph method. Why?