An Introduction to Spreadsheet Optimization Using Excel Solver

Download the latest version of this document and example files here:

http://www.meiss.com/en/publications/

Written by Joern Meissner and Thanh-Ha Nguyen. This tutorial is inspired by a similar document by Mark Broadie at Columbia Business School. Thanks for Sergei Savin for help and feedback.

1. Introduction

The major PC-based spreadsheets used today, Lotus, Excel and Quatro, all have built-in optimizers. These optimizers allow linear, non-linear, and integer programs to be solved within the spreadsheet. This guide provides a brief introduction to using the optimizer Solver, which is available in the *Analysis* group on the *Data* tab in Excel.

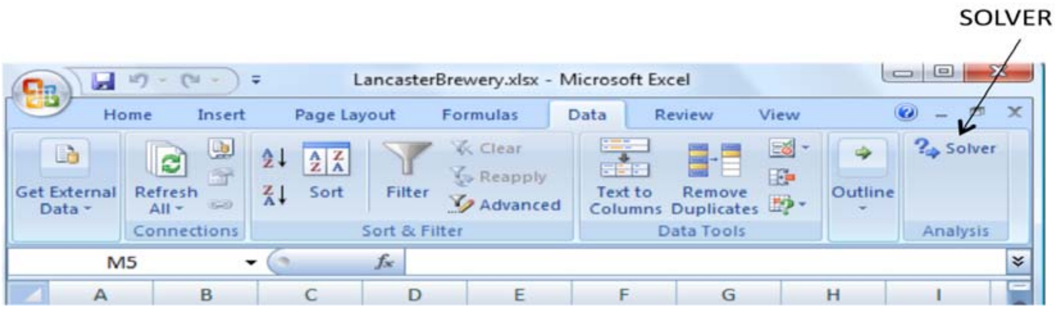


Figure 1: Solver in Excel

If the *Data* tab does not have the choice *Solver* available, then look at the *Office Button | Excel Options* to see if the *Add-Ins* option appear there. If it does appear, please select *Excel Add-ins* from the *Manage* drop down. Then click *Go...* and make sure that Solver is enabled. If the *Add-Ins* option does not appear, run the Setup program again to install it.

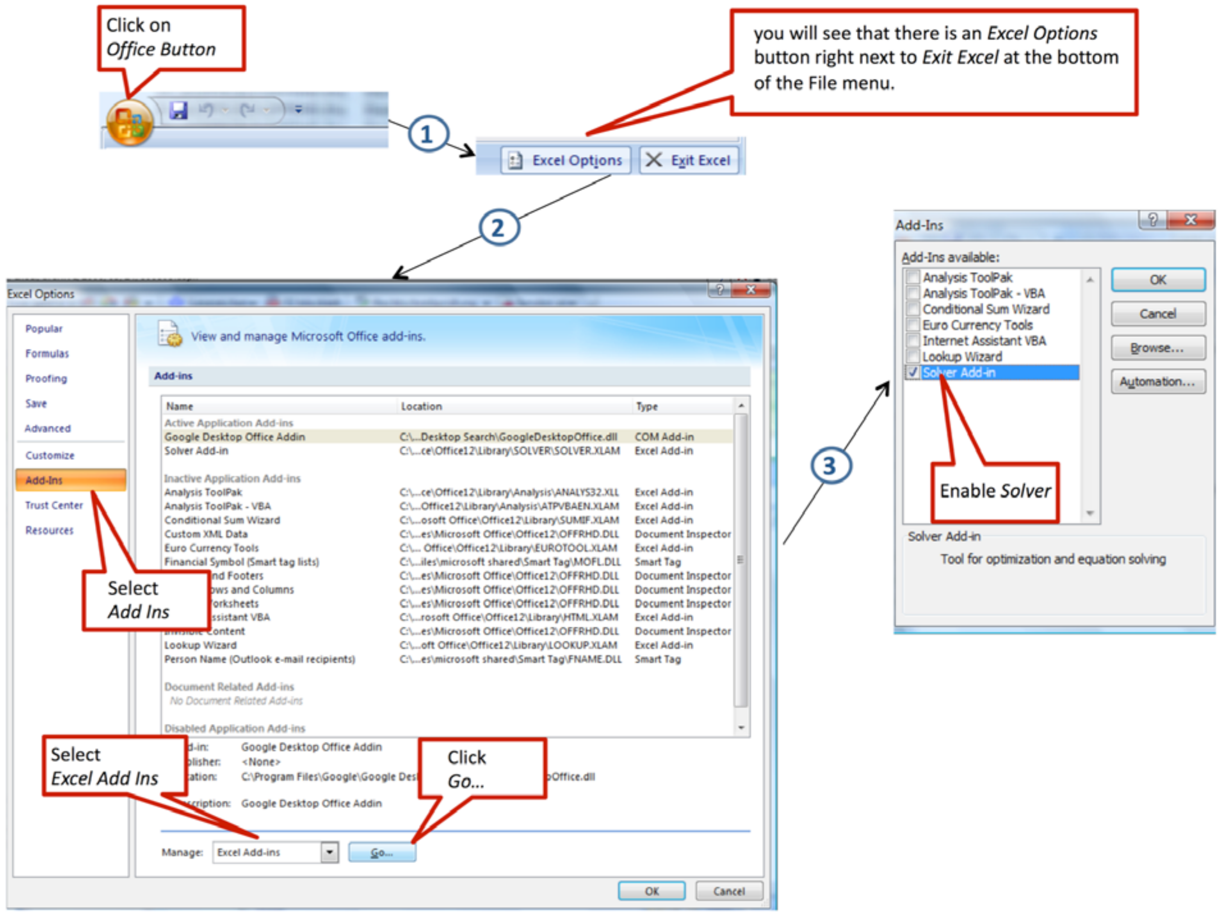


Figure 2: Enabling Solver

One advantage of the spreadsheet approach to optimization is that many optimization models can be represented in an understandable fashion in a spreadsheet. Another benefit is that many people are already familiar with using spreadsheets. The spreadsheet copy command allows large models with many similar constraints to be created and solved quickly in a spreadsheet environment.

Solving a linear program in Excel requires five main steps:

1. Creating a spreadsheet which models the problem.
2. Specifying the cell which contains the objective function.
3. Specifying the decision variables.
4. Specifying the cells which define the constraints.
5. Solving the model, i.e., optimizing.

Each step in the process is described in the following sections. All figures pertain to Microsoft Excel Version 12 for Windows 2007.

2. Creating a spreadsheet model

We will use a product mix example to illustrate how to implement a linear program in Excel. The Lancaster Brewery must decide how many barrels of the two types of beer (Lancaster Blonde and Lancaster Red) to produce for its next shipment. It has a limited supply of ingredients that are used in both products. Each barrel of Lancaster Blonde requires 20 lbs. of corn, 2 ozs. of hops, 10 lbs. of malt, and yields £20 of profit. Each barrel of Lancaster Red requires 1 oz. of hops, 30 lbs. of malt, and yields £30 of profit. The company has 500 lbs. of corn, 60 ozs. of hops, and 900 lbs. of malt on hand. Management wishes to determine the amount of Blonde and Red produce to maximize profits.

To answer this question, we need to create a spreadsheet which will calculate the profit and the amount of ingredients used for any amount of blonde beer and red beer produced. Figure 3 shows the beginning of the spreadsheet *LancasterBrewery.xlsx* that accomplishes this. All of the cell entries shown in the spreadsheet in Figure 3 are numerical values, i.e., no formulas have been entered yet. The cells C4 and D4 contain the amounts of Lancaster Red and Lancaster Blonde to be produced in the coming planning period. The profit contributions for each product are contained in the cells C5 and D5. The amount of each ingredient required per unit of each product is given in cells C10:D12. The quantity of ingredients available in the coming period is stored in cells G10:G12.

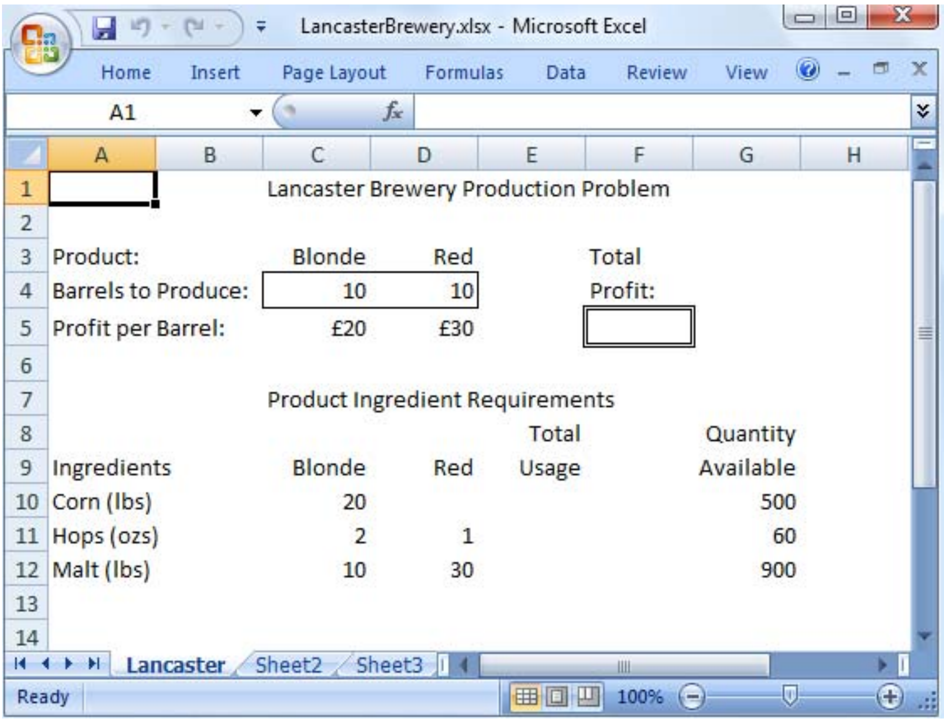


Figure 3: The preliminary spreadsheet *LancasterBrewery.xlsx*

The cells that we will designate as *decision variables*, cells C4 and C5, are highlighted with a box. The cell that we will designate as the *objective function*, cell F5, is highlighted with a double box. These boxes have no function in the spreadsheet except for style; they make the spreadsheet easier to read.

The next step is to create a formula for the profit in cell F5. Profit is the amount of Blonde produced multiplied by the profit per barrel of Blonde, plus the amount of Red produced multiplied by the profit per barrel of Red. The formula for profit in cell F5 is = SUMPRODUCT(C4:D4; C5:D5).[1]

---

[1] The function = SUMPRODUCT*(Block1; Block2)* multiplies each corresponding cell from *Block1* and *Block2* and then adds the results. For example, the formula =SUMPRODUCT(C4:D4; C5:D5) is equivalent to

The amount of corn required for a given production plan is computed in cell E10. The amount of corn required is the amount of Blonde produced (cell C4) times the corn requirement per barrel of Blonde (cell C10), plus the amount of Red produced (cell D4) times the corn requirement per barrel of Red (cell D10). The formula in E10 is = SUMPRODUCT($C$4:$D$4; C10:D10). This is equivalent to the formula +$C$4*C10+$D$4*D10. The dollar signs are added in the cell addresses for the production of Blonde and Red so that the formula in cell E10 can be correctly copied to cells E11 and E12[2]. The quantity of ingredients available in the coming period is stored in cells G10:G12.

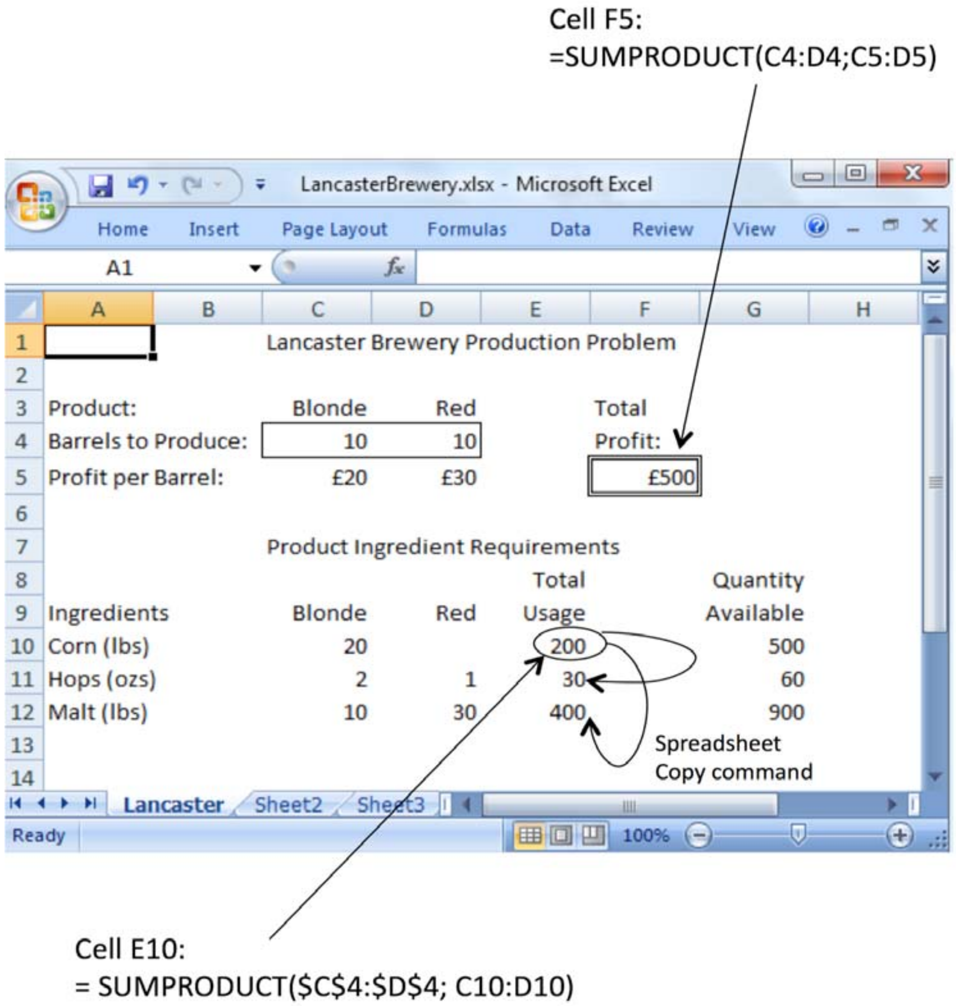


Figure 4: The spreadsheet *LancasterBrewery.xlsx* with formulas

The spreadsheet with these formulas is shown in Figure 4. We can try a few values for the production quantities to test the spreadsheet. For example, enter the production plan of 10 barrels

---

=C4*C5+D4*D5. In general, both blocks must be of the exact same size, i.e., the same number of rows and the same number of columns. For example, =SUMPRODUCT(A1:A5; B11:B15) is a valid formula while SUMPRODUCT(A1:A5; B1:B3) is not valid. The SUMPRODUCT function is especially useful if the blocks are large.

[2] The dollars signs in $C$4:$D$4 are used to fix the range when the formula is copied. Without the dollar signs, the formula in cell E11 (when copied from E10) would incorrectly read =SUMPRODUCT(C5:D5, C11:D11). With the dollar signs, the formula is copied correctly to cell E11 as =SUMPRODUCT($C$4:$D$4, C11:D11).

of Blonde (in cell C4) and 10 barrels of Red (in cell D4) to see what the results are. The total profit should be £500. The amount of corn used is 200 lbs., compared to the 500 lbs. that are available. 30 ozs. of hops are used out of 60. The production plan uses 400 lbs. of malt out of 900 lbs.

3. Specifying the objective function

With all of the formulas entered in the spreadsheet, the next step is to invoke the optimizer i.e. Solver which is available in the **Analysis** group on the **Data** tab (cf. Introduction).

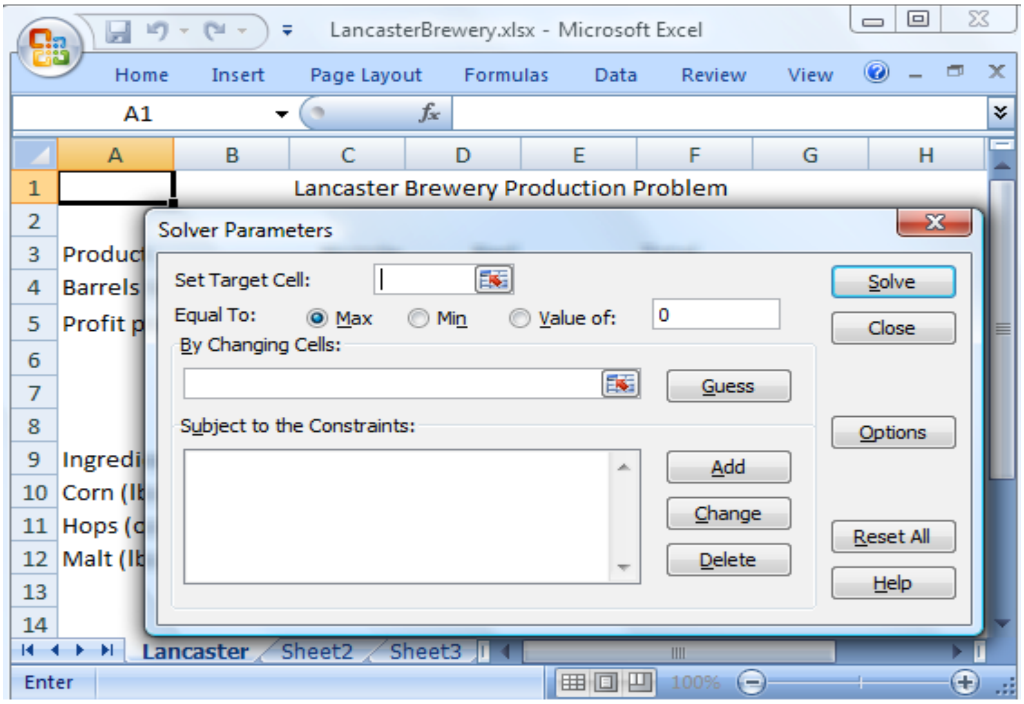


Figure 5: The *Solver Parameters* dialog box

This brings up the *Solver Parameters* dialog box shown in Figure 5. Notice that the cursor is initially positioned in the box "Set Target Cell," which is Excel's term for "objective function". For Lancaster Brewery, the objective is to maximize total profit, which is computed in cell F5. Type F5 or use the mouse to highlight cell F5. Note that the target cell is specified as "Max" by default. For a minimization problem, the circle "Min" should be checked.

Do not hit <Enter> at this point, because that is equivalent to asking the optimizer to solve the problem. Before doing that, the decision variables and constraints need to be specified.

4. Specifying the decision variables

In our example, the amounts of beer and ale to produce are the *decision variables*, located in cells C4:D4. The decision variable are called *changing cells* in Excel. Use the mouse or type <Alt>-<B> to move the cursor to the box labeled "By Changing Cells" in the *Solver Parameters* dialog box. Then type C4:D4 for use the mouse to highlight the block C4:D4.

Additional decision variables in different locations of the spreadsheet can be entered using commas as separators. For example, you could type C4:D4, C10:D12, A14:E16 in the "By Changing

Cells" box. In our case this is not necessary, since there are only two decision variables (cells C4 and D4).

5. Specifying the constraints and the model type

The next step is to specify the constraints. We first specify the nonnegativity constraints. That is, we need to guarantee that the optimal values in cells C4:D4 are nonnegative. Use the mouse to click on the "Add" button or type <Alt>-<A>. This brings up the *Add Constraint* dialog box shown in Figure 4 with the cursor initially positioned in the "Cell Reference" box.

In the "Cell Reference" part of the *Add Constraint* dialog box, type C4:D4 or use the mouse to highlight cells C4:D4. Next select the down arrow and choose ">=". Then enter a zero in the "Constraint" box. This can be done from the keyboard by typing <Alt>-<C> and then 0, or by using the mouse to move to the appropriate area and then typing 0. This method adds two nonnegativity constraints at one time, C4 >= 0 and D4 >= 0, represented as $C$4:$D$4 >= 0. Since more constraints need to be added, type <Alt>-<A> or use the mouse to click on the "Add" button.

The only other constraints are that the total usage of corn, hops, and malt must be less than or equal to the corresponding quantities available. In the "Cell Reference" part of the Add Constraint dialog box, type E10:E12 or use the mouse to highlight cells E10:E12. The "<=" constraint is the default. Now in the "Constraint" box enter G10:G12. This can be done from the keyboard by typing <Alt>-<C> and then typing G10:G12, or by using the mouse to click in the "Constraint" box and then highlighting cells G10:G12 in the spreadsheet. This procedure has specified three separate constraints at once. Since no more constraints need to be added, click the "OK" button. The Solver Parameters dialog box should return as shown in Figure 6.

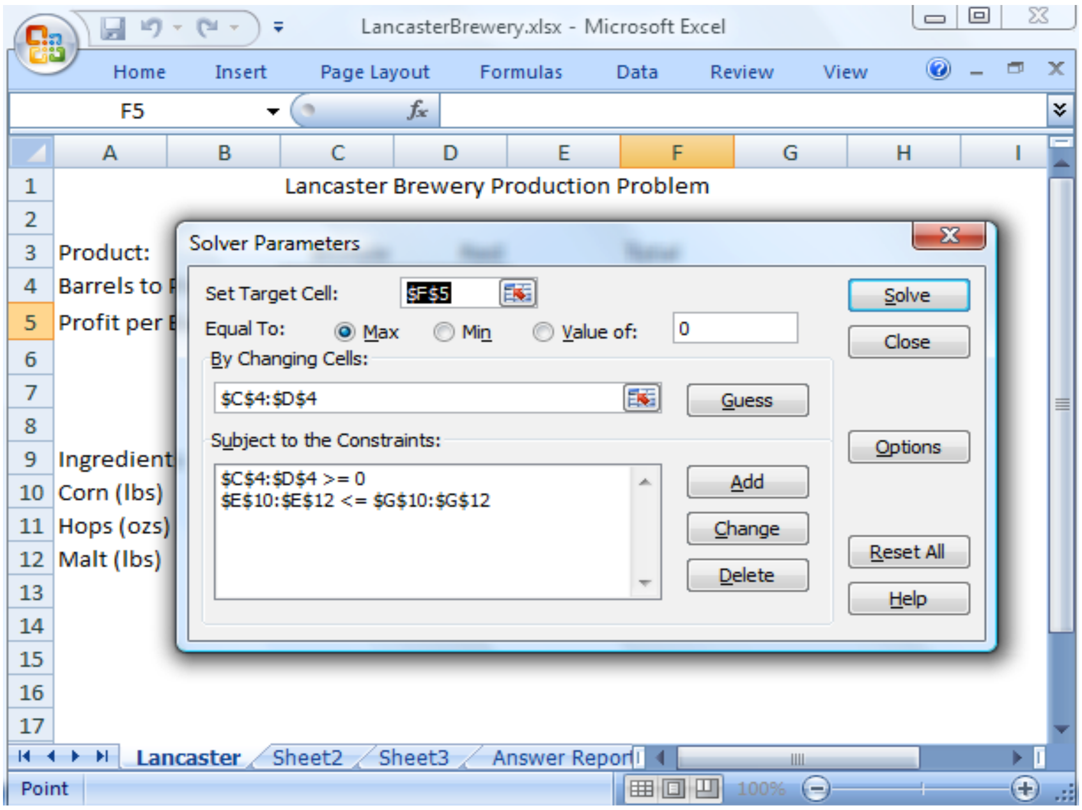


Figure 6: The *Solver Parameters* dialog box with constraints added

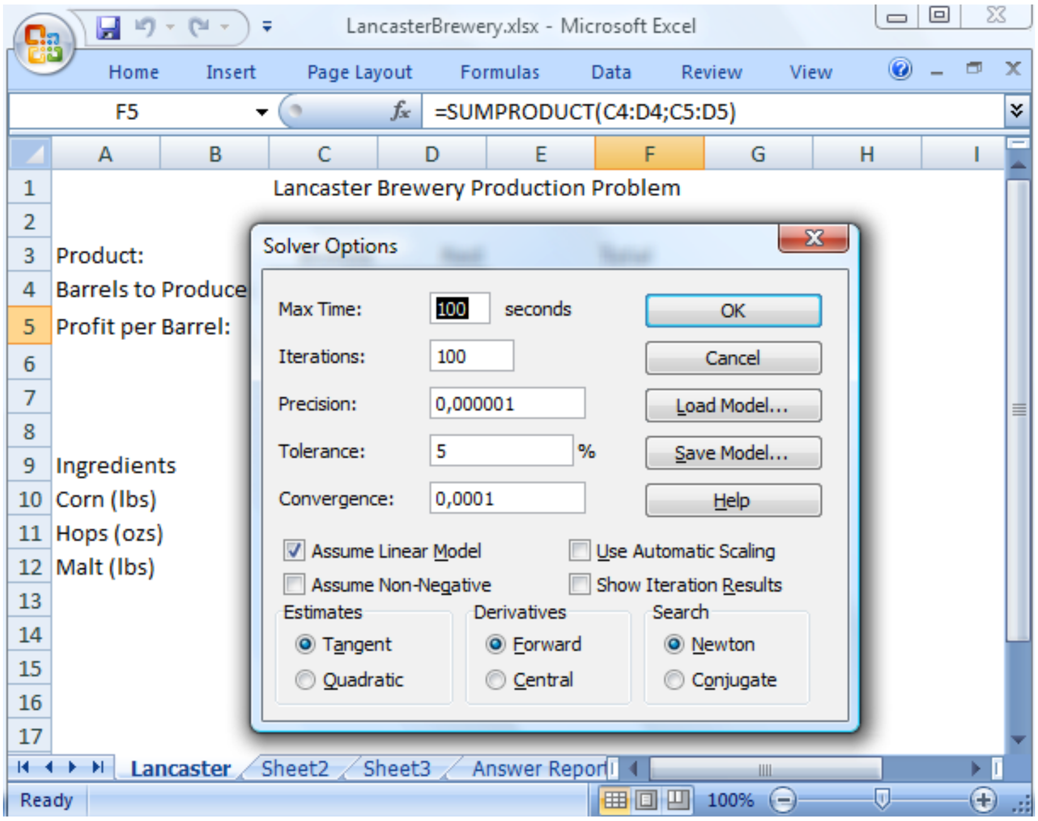
In the “Subject to the Constraints” part of the *Solver Parameters* dialog box there should now be two lines. The first line $C$4:$D$4 >=0 is short for the two constraints C4 >= 0 and D4 >= 0. Similarly, the second line $E$10:$E$12 <= $G$10:$G$12 is short for the tree constraints E10 <= G10, E11 <= G11, and E12 <= G12. Constraints can be deleted by highlighting the constraint and then hitting the “Delete” button, or edited using the “Change” button in the *Solver Parameters* dialog box.

This linear program has two decision variables and five constraints. Excel version 12 allows at most 200 decision variables. It allows one upper limit and one lower limit constraint on any decision variable plus up to 100 additional constraints[3].

<u>Specifying a linear model</u>

Since this model is a linear program (as opposed to a nonlinear program), it is good practice to let the optimizer know this. Choose the “Options” button in the *Solver Parameters* dialog box. This brings up the *Solver Options* dialog box shown in Figure 7.

Check the “Assume Linear Model” box and then click on the “OK” button. This tells the optimizer that the model is a linear program, as opposed to a more complicated nonlinear program. The advantages of specifying a linear model are faster solution times, more accurate and reliable results, and the availability of more detailed sensitivity analysis reports. After clicking on the “OK” button the *Solver Parameters* dialog box should return.



[3] http://www.solver.com/suppstdsizelim.htm

Figure 7: The *Solver Options* dialog box

6. Solving the model

The production model for the Lancaster Brewery is now ready to be solved. To activate the optimizer, type <Alt>-<S> or use the mouse to click the "Solve" button. After a few seconds, the *Solver Results* dialog box should appear. Since the default is "Keep Solver Solution" clicking on "OK" will close the dialog box and reveal the optimal solution as illustrated in Figure 8. We will return to the "Reports" part of the *Solver Results* dialog box a bit later.

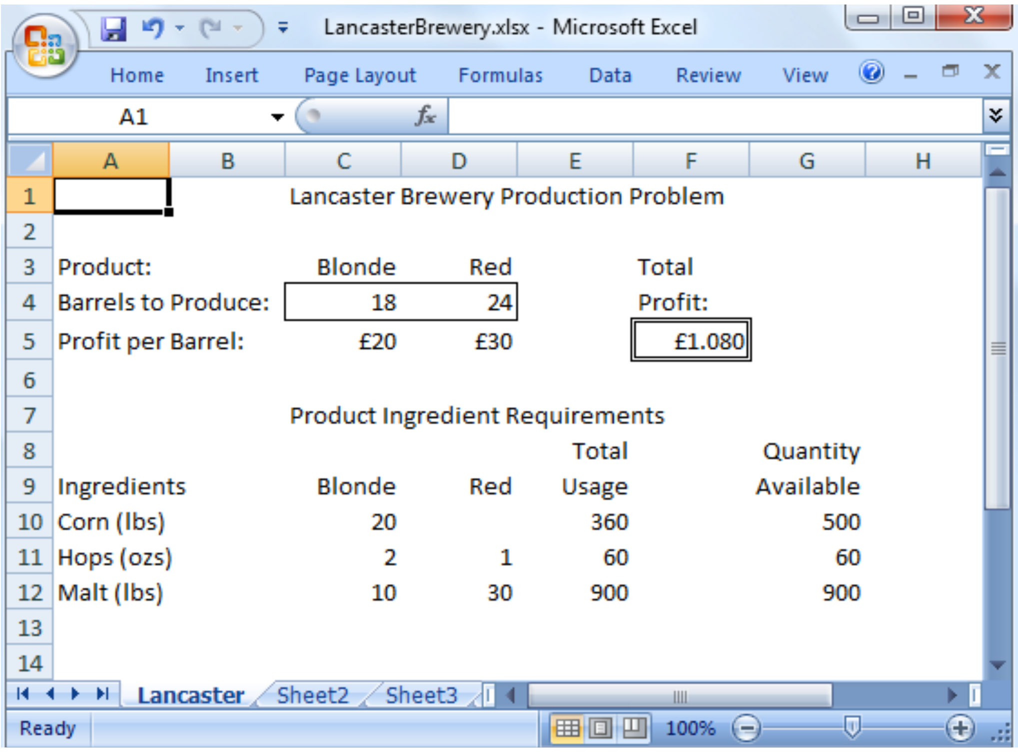


Figure 8: The spreadsheet after optimizing

7. Spreadsheet style

One of the drawbacks of spreadsheets is a *lack of documentation*. For example, the numerical values for the optimal solution are clearly shown in Figure 8. However, none of the formulas used in the model are displayed. None of the constraints are visible in Figure 8. To aid in communicating the underlying ideas in a spreadsheet it is important to adopt good spreadsheet *style*. While style is somewhat a matter of personal taste, there's good taste and bad taste.

Throughout this course, we will adopt some stylistic conventions. First, all decision variables will be enclosed in a *single* box and (for those fortunate enough to have color displays) colored *blue*. The objective function will be enclosed in a *double* box and colored *red*. Constraints will be indicated using =IF functions as described below.

To put a box around a block of decision variables, first highlight (select) the block. Then click on the right mouse button to bring up a menu. Move the cursor to the "Format Cells…" menu item and click on the left mouse button. The *Format Cells* dialog box should appear. Choose the “Border” tab and in the “Style” section choose the single line (which is the default). Then choose the border “Outline”. Finally, click on the “OK” button for the line drawing to take effect.

To change the text color of the decision variables, bring up the *Format Cells* dialog box as before. Choose the “Font” tab, select “Color”, select the color blue, and then click on the “OK” button.

The same procedure can be used to put a double box around the objective function cell (The difference is that a double line should be selected within line “Style”). Then change the text color of the objective cell to red.

<u>*Constraint style*</u>

To view the constraints the user can always bring up the *Solver Parameters* dialog box. The user then has to decipher the line $E$10:$E$12 <= $G$10:$G$12, which really means that the amount of each ingredient used must be less than or equal to the corresponding quantity available. As an alternative, consider putting the formula

=IF(C10<=G10;"<=";"Not <=")

in cell F10. This formula says that if the amount of corn used is less than or equal to the amount available, display “<=”, otherwise, display “Not<=”. This formula serves two purposes. First, it gives a quick visual indication on the spreadsheet that a constraint exists. Second, it indicates whether or not the constraint is violated.

Unfortunately, because computers work with finite precision numbers, the previous formula does not always work. For example, after optimizing cell E10 could contain the value 500.000000001, which is greater than 500 stored in cell G10. To the computer, this value is “close enough” to satisfying the constraint E10 <= G10. To accommodate this event, the formula placed in cell F10 is

=IF(E10<=G10+1.0E-05;"<=";"Not <=")

(Recall 1.0E-05 means 0,00001). Now, even if the value in cell F10 is a tiny bit greater than the value in cell G10, cell F10 will show “<=”. The formula in cell F10 is called a *constraint indicator*. It can be copied to cells F11:F12 to indicate the two other scarce resource constraints.

Figure 9 shows what the spreadsheet looks like for the production plan consisting of 50 barrels of Red only. The malt constraint is violated because the production plan requires 1,500 lbs. of corn, but only 900 lbs. are available. Cell F12 displays “Not <=” and the production plan is *infeasible*. The corn and hops constraints are not violated, so cells F10 and F11 both display “<=”. For readability in the spreadsheet, it is a good idea, where possible, to keep a left-to-right constraint order: constrained cell, constraint indicator, constraint limit.

Suppose the model had a greater than or equal to constraint, e.g., E10 >= G10. Then constraint indicator in cell F10 would be

=IF(E10<=G10+1.0E-05;"<=";"Not <=")

For an equality constraint, e.g., E10 = G10, the constraint indicator in cell F10 would be

=IF(ABS(E10-G10)<1.0E-05;"=";"Not =").

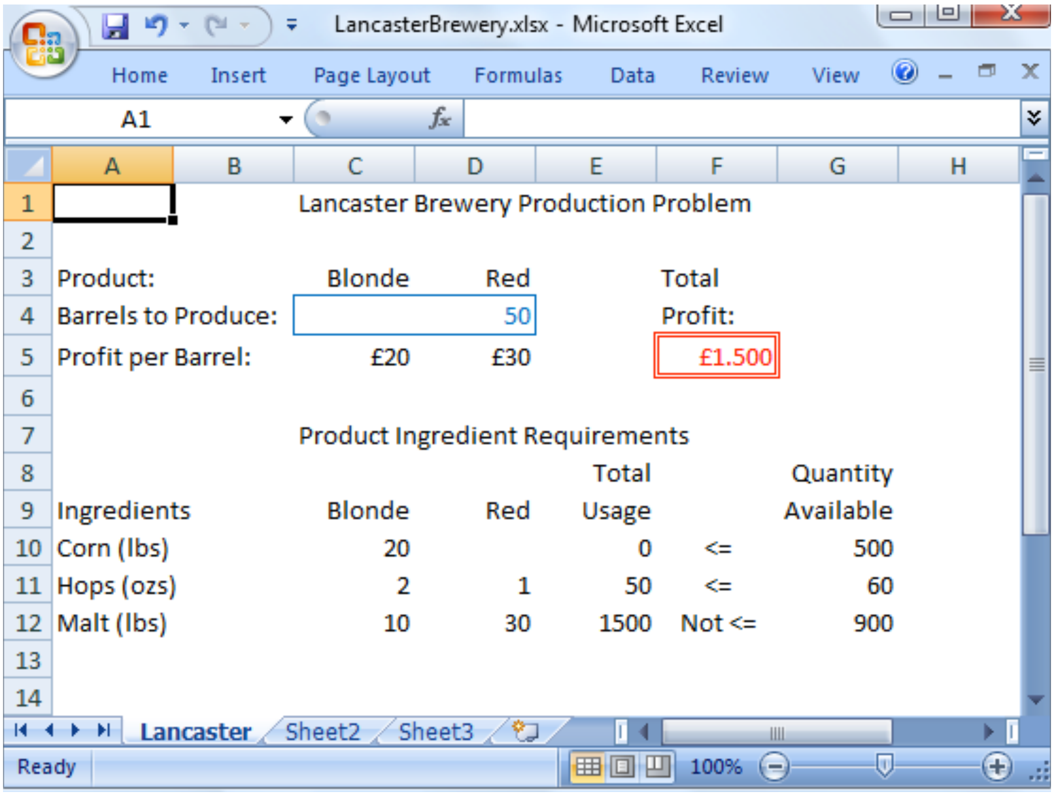


Figure 9: The spreadsheet with constraints indicated

8. Warning messages and potential problem areas

There are several pitfalls to avoid when using the Excel for Window optimizer. For example, spreadsheet models often need to be scaled so that there are not extremely large and extremely small numbers in the same model. This issue and other potential problems are treated in this section.

Selected data doesn't qualify as a linear model

One error message that the Excel optimizer can give is "The conditions for Assume Linear Model are not satisfied". This can happen if the model is nonlinear, but the user checked the "Assume Linear Model" option. However, it can also happen because the problem is poorly scaled. A poorly scaled model is one with numbers that are very different in size, e.g. 0.0003 and 400,000,000. Often the units of the decision variables and/or constraints can be changed to achieve a better "re-scaled" model. Alternatively, the Precision of the optimizer can be changed in the *Solver Options* dialog box. For example, the precision could be changed from 1.0E-06 to 0.001.

Model too large

Unfortunately Excel has built-in limits on the size of models it can solve. The current version limits optimization models to at most 200 decision variables. It allows one upper limit and one lower limit constraint on any decision variable plus up to 100 additional constraints[4].

Difficult nonlinear programs

After hitting the "Solve" button in the *Solver Parameters* dialog box, the optimizer usually returns with the *Solver Results* dialog box and the message "Solver found a solution. All constraints and optimality conditions are satisfied". For some highly nonlinear problems, the optimizer might not give the optimal solution, even though it says otherwise. If you suspect this is the case, try changing the initial values of the decision variables o a better guess and then re-solve the model. Other possible remedies include changing some of the other options in the *Solver Options* dialog box. If the model is poorly scaled, try changing the units of the decision variables and/or constraints to achieve a better "re-scaled" model.

9. Specifying integer decision variables

The Excel optimizer can solve optimization models with integer variables. In Excel, an integer variable refers to a general integer variable, i.e. a variable restricted to the values …, -2, -1, 0, 1, 2, … Integer variables can be specified in the *Add Constraint* dialog box by selecting the down arrow and choosing "int".

To specify a binary (0, 1) variable, first constrain the variable to be integer as described above. Then add two constraints specifying that the variable must be nonnegative and less than or equal to 1.

Be sparing in the use of integer variables, because they can significantly slow the optimizer. It almost never necessary to restrict variables representing the number of items to produce to be integer. For example, suppose that a decision variable represents the number of cars to produce in a year. Does it really matter that the solution is to produce 314,234.8 cars? The input data is probably not accurate to more than two significant digits, so it does not make sense to worry about the fractional 0.8 car.

The optimizer does not always solve integer programs to optimality. The *tolerance* parameter in the *Solver Options* dialog box is set to 5% by default. This means that the optimizer will stop when the solution within 5% of the true optimal solution. Even with the tolerance parameter set to 5% the optimizer will often give the true optimal solution. The default setting is 0.05 because integer programs can take a very long time to solve. To ensure that the exact optimal solution is obtained, the tolerance parameter shout be set to 0.0%.

[4] http://www.solver.com/suppstdsizelim.htm

10. Report files and dual prices

Excel can generate more detailed solution reports, rather than just showing the optimal solution in the spreadsheet. Just after solving a model, the *Solver Results* dialog box should appear. The "Reports" part of the Solver Results dialog box contains three options: "Answer", "Sensitivity" and "Limits". For our purposes, the most useful of these are the "Answer" and "Sensitivity" reports. Select these reports and then click on "OK". Excel will then create new sheets in the workbook called Answer Report 1 and Sensitivity Report 1. The answer report is shown in Figure 10 and the sensitivity report is shown in Figure 11.

Answer Report1

Microsoft Excel 12.0 Answer Report
Worksheet: [LancasterBrewery.xlsx]Lancaster
Report Created: 15.12.2008 10:57:29

Target Cell (Max)

Cell	Name	Original Value	Final Value
$F$5	Profit per Barrel: Profit:	£1.080	£1.080

Adjustable Cells

Cell	Name	Original Value	Final Value
$C$4	Barrels to Produce: Blonde	18	18
$D$4	Barrels to Produce: Red	24	24

Constraints

Cell	Name	Cell Value	Formula	Status	Slack
$E$10	Corn (lbs) Usage	360	$E$10<=$G$10	Not Binding	140
$E$11	Hops (ozs) Usage	60	$E$11<=$G$11	Binding	0
$E$12	Malt (lbs) Usage	900	$E$12<=$G$12	Binding	0
$C$4	Barrels to Produce: Blonde	18	$C$4>=0	Not Binding	18
$D$4	Barrels to Produce: Red	24	$D$4>=0	Not Binding	24

Figure 10: Answer report for the spreadsheet *LancasterBrewery.xlsx*

Sensitivity report1

Microsoft Excel 12.0 Sensitivity Report
Worksheet: [LancasterBrewery.xlsx]Lancaster
Report Created: 15.12.2008 10:57:42

Adjustable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
$C$4	Barrels to Produce: Blonde	18	0	20	40	10
$D$4	Barrels to Produce: Red	24	0	30	30	20

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
$E$10	Corn (lbs) Usage	360	0	500	1E+30	140
$E$11	Hops (ozs) Usage	60	6	60	11,66666667	30
$E$12	Malt (lbs) Usage	900	1	900	900	350

Figure 11: Sensitivity report for the spreadsheet *LancasterBrewery.xlsx*

The size of the reports depends on the number of decision variables and constraints in the model. In the answer report, information is given indicating whether or not constraints are binding at the optimal solution. In addition, the answer report gives slack values. The sensitivity report gives shadow prices (i.e. dual values), right-hand side ranges and other information. For example, the dual value for the hops constraint (6, in this case) represents the change in the optimal objective function value per unit change in the quantity of hops available. If 61 ozs. of hops were available the optimal objective profit would increase to £1,086.

11. Saving and loading optimization models

Excel saves the most recent optimization model in the spreadsheet. So saving a model is *not* really necessary unless you need to keep multiple optimization models in a single workbook (i.e. spreadsheet). For example, it is sometimes useful to keep models with different sets of constraints or different objective functions. Saving and loading models can also be handy in conjunction with spreadsheet macros.

Saving and loading optimization models is done in the Solver Options dialog box. Choose *Data|Solver|Options* to bring up the *Solver Options* dialog box. Then choose “Save Model” to save an optimization model or “Load Model” to load an optimization model. Additional information is available from the “Help” button. It is a good practice to save the model to another sheet of the workbook.