2

Solving Math Programs with LINGO

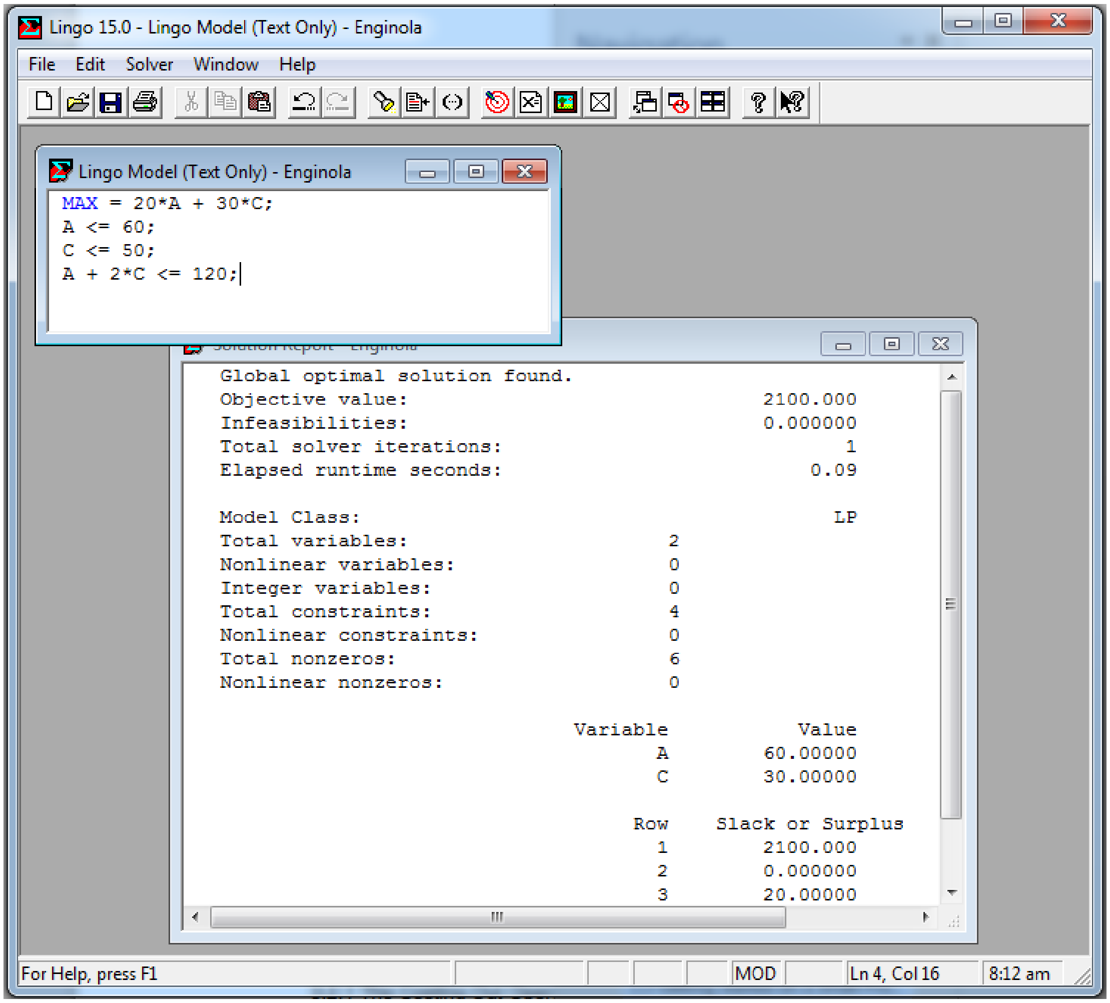
2.1 Introduction

The process of solving a math program requires a large number of calculations and is, therefore, best performed by a computer program. The computer program we will use is called LINGO. The main purpose of LINGO is to allow a user to quickly input a model formulation, solve it, assess the correctness or appropriateness of the formulation based on the solution, quickly make minor modifications to the formulation, and repeat the process. LINGO features a wide range of commands, any of which may be invoked at any time. LINGO checks whether a particular command makes sense in a particular context.

The main version of LINGO has a graphical user interface(GUI), althought there is a command line interface available for certain special situations, e.g. running under Unix. We will work only with the GUI version.

2.2 LINGO for Windows, Apple Mac, and Linux

When the GUI version LINGO starts, it opens a blank window known as a *Model Window*. The Model Window is where you "do all your work". Output in LINGO is displayed in a *Report Window*. LINGO can generate a number of reports pertaining to your model. All the standard commands for opening and saving files, familiar to Windows and Mac users are available. The following is a typical screen shot.


Lingo 15.0 - Lingo Model (Text Only) - Enginola
File Edit Solver Window Help
Lingo Model (Text Only) - Enginola
For Help, press F1
MOD
Ln 4, Col 16
8:12 am

```
MAX = 20*A + 30*C;
A <= 60;
C <= 50;
A + 2*C <= 120;
```

```
Global optimal solution found.
Objective value:                              2100.000
Infeasibilities:                              0.000000
Total solver iterations:                             1
Elapsed runtime seconds:                          0.09

Model Class:                                        LP

Total variables:                      2
Nonlinear variables:                  0
Integer variables:                    0

Total constraints:                    4
Nonlinear constraints:                0

Total nonzeros:                       6
Nonlinear nonzeros:                   0

                              Variable           Value
                                     A        60.00000
                                     C        30.00000

                                   Row    Slack or Surplus
                                     1        2100.000
                                     2        0.000000
                                     3        20.00000
```

Some of the less common commands available in the GUI version of LINGO are:

SOLVE

Use the *SOLVE* command from the *LINGO/Solver* menu, click on the button, or press *Ctrl+U* to send the model currently in memory to the LINGO solver. If you have more than one model open, the frontmost (or active) window is the one in memory.

MATCH PARENTHESIS Ctrl+P

Use the *MATCH PARENTHESIS* command from the *Edit* menu, click the button, or type *Ctrl+P* to find the close parenthesis that corresponds to the open parenthesis you have selected.

In addition to this command, there is one other way to find matching parentheses. LINGO will highlight matching parentheses in red when the *Match Paren* option is enabled under the *LINGO|Options* command (see below). By placing the cursor immediately after one of the parentheses of interest, you will notice that the color of the parenthesis changes from black to red. LINGO will simultaneously display the matching parenthesis in red. These parentheses will remain displayed in red until you move the cursor to another position.

PASTE FUNCTION

Use the *PASTE FUNCTION* command from the *Edit* menu to paste any of LINGO's built-in functions at the current insertion point. Choose the category of the LINGO function you want to paste, then select the function from the cascading menu. LINGO inserts place holders for arguments in the functions.

SELECT FONT... Ctrl +J

Use the *SELECT FONT* command from the *Edit* menu or press *Ctrl+J* to select a new font in which to display the currently selected text.

INSERT NEW OBJECT

Use the *INSERT NEW OBJECT* command from the *Edit* menu to embed an OLE object into the LINGO document.

LINKS

Use the *LINKS* command from the *Edit* menu to control the links to external objects in your document.

OBJECT PROPERTIES Alt+Enter

Use the *OBJECT PROPERTIES* command from the *Edit* menu or press *Alt+Enter* to specify the properties of a selected, embedded object

2.2.1 LINGO Menu

SOLUTION… X= Ctrl+W

Use the *SOLUTION* command from the *LINGO* menu, click the button, or press *Ctrl+W* to open the Solutions dialog box. Here you can specify the way you want a report of the solution currently in memory to appear. When you click OK, LINGO writes the report to a Report Window.

GENERATE… Ctrl+G/Ctrl+Q

Use the *DISPLAY MODEL* and *DON'T DISPLAY MODEL* sub-commands from the *LINGO Solver | Generate* command or press *Ctrl+G* or Ctrl+Q, respectively, to create an expanded version of the current model. The expanded model explicitly lists all the generated constraints and variables in your model.

If you choose to display the model, LINGO will place a copy of the generated model in a new window, which you may scroll through to examine, print, or save to disk. If you choose not to display the model, LINDO will generate the model without displaying it, but will store the generated model for later use by the appropriate solver.

PICTURE Ctrl+K
Use the *PICTURE* command from the *LINGO* menu or press *Ctrl+K* to display a model in matrix form. Viewing the model in matrix form can be helpful in identifying special structure in your model.

2.2.2 Windows Menu

COMMAND WINDOW Ctrl +1
Use the *COMMAND WINDOW* command from the *Windows* menu or press *Ctrl+1* to open LINGO's Command Window. The Command Window gives you access to LINGO's command line interface. In general, Windows users will not need to make use of the Command Window. It is provided for users who may want to put together application-specific "products" that make use of LINGO through Command Window scripts to control the program. Please refer to your help file or user's manual for more information on the command line commands.

STATUS WINDOW Ctrl +2

Use the *STATUS WINDOW* command from the *Windows* menu or press *Ctrl+2* to open LINGO's Solver Status window.

2.2.3 Help Menu

HELP TOPICS
Use the *HELP TOPICS* command from the *Help* menu, or click on the first question mark button to open LINGO help to the Contents section. Press the second button (with the arrow) to invoke context-sensitive help. Once the cursor has changed to the question mark, selecting any command will take you to help for that command.

REGISTER

Use the *REGISTER* command from the *Help* menu to register your version of LINGO online. You will need a connection to the internet open for this command to work. Enter your personal information in the dialog box supplied and select the register button. Your information will be sent directly to LINDO Systems via the Internet.

LINDO Systems is constantly working to make our products faster and easier to use. Registering your software with LINDO ensures that you will be kept up-to-date on the latest enhancements and other product news.

AUTOUPDATE

Use the *AUTOUPDATE* command from the *Help* menu to have LINGO automatically check every time you start the LINGO software whether there is a more recent version of LINGO available for download on the LINDO Systems website. You will need a connection to the internet open for this command to work.

ABOUT LINGO…

Use the *ABOUT LINGO* command from the *Help* menu to view information about the version of LINGO you are currently using (e.g., the release number, constraint limit, variable limit, and memory limit).

2.2.4 Summary

This is not intended to be an exhaustive description of the commands available in the Windows version of LINGO. Please refer to your help file or user's manual for a more in-depth analysis.

2.3 Getting Started on a Small Problem

When you start LINGO for Windows, the program opens an <*untitled*> window for you. For purposes of introduction, let's enter the Enginola problem we looked at in the previous chapter directly into this <*untitled*> window:

```
MAX = (20 * A) + (30 * C);
!note that the parentheses aren't needed, because LINGO;
!will do multiplication and division first;
A < 60;
C < 50;
A + 2 * C < 120;
```

Note, even though the strict inequality, "<", was entered above, LINGO interprets it as the loose inequality, "≤". The reason is that typical keyboards have only the strict inequalities, < and >. You may, and in fact are encouraged to, use the two symbols "<=" to emphasize an inequality is of a less-than-or-equal-to nature. Also, notice comments are preceded by the exclamation mark (!). A semicolon (;) terminates a comment.

Click on the Solve/"bullseye" button, use the *Solve* command from the *Solve* menu, or press *Ctrl+U* to solve the model. While solving, LINGO will show the Solver Status Window with information about the model and the solution process. When it's done solving, the "State" field should read "Global Optimum". Then, click on the "Close" button to close the Solver Status Window:

The following solution is now in a Report Window:

```
Optimal solution found at step:          1
Objective value:                  2100.000
Variable           Value        Reduced Cost
       A        60.00000           0.0000000
       C        30.00000           0.0000000
     Row    Slack or Surplus      Dual Price
       1        2100.000            1.000000
       2       0.0000000            5.000000
       3        20.00000           0.0000000
       4       0.0000000            15.00000
```

Editing the model is simply a matter of finding and changing the variable, coefficient, or direction you want to change. Any changes will be taken into account the next time that you solve the model.

Click on the button, use the *Save* command from the *File* menu, or press *Ctrl+S* to save your work.

2.4 Integer Programming with LINGO

Fairly shortly after you start looking at problems for which optimization might be applicable, you discover the need to restrict certain variables to integer values (i.e., 0, 1, 2, etc.). LINGO allows you to identify such variables. We give an introductory treatment here. It is discussed more thoroughly in Chapter 11, *Formulating and Solving Integer Programs*. Integer variables in LINGO can be either 0/1 or general. Variables restricted to the values 0 or 1 are identified with the *@BIN* specification. Variables that may be 0, 1, 2, etc., are identified with the *@GIN* specification.

In the following model, the variables *TOM*, *DICK*, and *HARRY* are restricted to be 0 or 1:

```
MAX = 4 * TOM  + 3 * DICK  + 2 * HARRY;
    2.5 * TOM            + 3.1 * HARRY <= 5;
     .2 * TOM + .7 * DICK + .4 * HARRY <= 1;
@BIN(TOM);
@BIN(DICK);
@BIN(HARRY);
```

After solving, to see the solution, choose *Solution* from the *Reports* menu, or click on the button, and choose *All Values*. The Report Window displays the following:

```
Optimal solution found at step:          1
Objective value:                  7.000000
Branch count:                            0
Variable           Value        Reduced Cost
     TOM        1.000000           -4.000000
    DICK        1.000000           -3.000000
   HARRY       0.0000000           -2.000000
     Row    Slack or Surplus      Dual Price
       1        7.000000            1.000000
       2        2.500000           0.0000000
       3       0.1000000           0.0000000
```

General integers, which can be 0, 1, 2, etc., are identified in analogous fashion by using *@GIN* instead of *@BIN*, for example:

```
@GIN(TONIC);
```

This restricts the variable *TONIC* to 0, 1, 2, 3, ….

The solution method used is branch-and-bound. It is an intelligent enumeration process that will find a sequence of better and better solutions. As each one is found, the Status Window will be updated with the objective value and a bound on how good a solution might still remain. After the enumeration is complete, various commands from the Reports menu can be used to reveal information about the best solution found.

Let's look at a slightly modified version of the original Enginola problem and see how the GIN specification might help:

```
MAX = 20 * A + 30 * C;
A < 60;
C < 50;
A + 2 * C < 115;
```

Notice the capacity of 115 on the labor constraint (Row 4):

```
Optimal solution found at step:          1
Objective value:                  2025.000

Variable           Value        Reduced Cost
       A        60.00000           0.0000000
       C        27.50000           0.0000000

     Row    Slack or Surplus      Dual Price
       1        2025.000            1.000000
       2       0.0000000            5.000000
       3        22.50000           0.0000000
       4       0.0000000            15.00000
```

Note that a fractional quantity is recommended for *C*. If fractional quantities are undesirable, declare *A* and *C* as general integer variables:

```
MAX = 20 * A + 30 * C;
A < 60;
C < 50;
A + 2 * C < 115;
@GIN( A);
@GIN( C);
```

Solving results in the following:

```
Optimal solution found at step:          4
Objective value:                  2020.000
Branch count:                            1

Variable           Value        Reduced Cost
       A        59.00000           -20.00000
       C        28.00000           -30.00000

     Row    Slack or Surplus      Dual Price
       1        2020.000            1.000000
       2        1.000000           0.0000000
       3        22.00000           0.0000000
       4       0.0000000           0.0000000
```

2.4.1 Warning for Integer Programs

Although the integer programming (IP) capability is very powerful, it requires skill to use effectively. In contrast to linear programs, just because you can formulate a problem as an integer program, does not mean that it can be solved in very little time. It is very easy to prepare a bad formulation for an essentially easy problem. A bad formulation may require intolerable amounts of computer time to solve. Therefore, you should have access to someone who is experienced in IP formulations if you plan to make use of the IP capability. Good formulations of integer programs are discussed further in Chapter 11, *Formulating and Solving Integer Programs*.

2.5 Solving an Optimization Model

Solving a linear or integer program is a numerically intensive process. We do not discuss the implementation details of the solution algorithms. Writing an efficient solver requires several person-years of effort. For a good introduction to some of the algorithms, see Martin (1999) or Greenberg (1978).

Even though commercial optimization is quite robust, good practice is to avoid using extremely small or extremely large numbers in a formulation. You should try to “scale” the model, so there are no extremely small or large numbers. You should not measure weight in ounces one place and volume in cubic miles somewhere else in the same problem). A rule of thumb is there should be no nonzero coefficient whose absolute value is greater than 100,000 or less than 0.0001. If LINGO feels the model is poorly scaled, it will display a warning. You can usually disregard this warning. However, it is good practice to choose your units of measure appropriately, so this message does not appear.

2.6 Problems

1. Recall the Enginola/Astro/Cosmo problem of the previous chapter. Suppose we add the restriction that only an even number (0, 2, 4…) of Cosmos are allowed. Show how to exploit the *@GIN* command to represent this feature. Note, this kind of restriction sometimes arises in the manufacture of plastic wrap. The product starts out as a long hollow tube. It is flattened and then two resulting edges are cut off to leave you with two flat pieces. Thus, the number of units produced is always a multiple of 2.
2. Using your favorite text editor, enter the Enginola formulation. Save it as a simple, unformatted text file. Start up LINGO, read the model into LINGO, and solve it.
3. Continuing from (2), use LINGO to prepare an output file containing both the formulation and solution. Read this file into your favorite text editor and print it.