### MATLAB and Optimization

### Optimization Decision Table

The following table is designed to help you choose a solver. It does not address multiobjective optimization or equation solving. There are more details on all the solvers in [Problems Handled by Optimization Toolbox Functions](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/brhkghv-18.html#brhkghv-21).

Use the table as follows:

1. Identify your objective function as one of five types:
   * Linear
   * Quadratic
   * Sum-of-squares (Least squares)
   * Smooth nonlinear
   * Nonsmooth
2. Identify your constraints as one of five types:
   * None (unconstrained)
   * Bound
   * Linear (including bound)
   * General smooth
   * Discrete (integer)
3. Use the table to identify a relevant solver.

In this table:

* Blank entries means there is no Optimization Toolbox solver specifically designed for this type of problem.
* \* means relevant solvers are found in [Global Optimization Toolbox](xprod:../../../toolbox/gads/brdvu23-1.html) functions (licensed separately from Optimization Toolbox solvers).
* fmincon applies to most smooth objective functions with smooth constraints. It is not listed as a preferred solver for least squares or linear or quadratic programming because the listed solvers are usually more efficient.
* The table has suggested functions, but it is not meant to unduly restrict your choices. For example, fmincon is known to be effective on some nonsmooth problems.
* The Global Optimization Toolbox ga function can be programmed to address discrete problems. It is not listed in the table because additional programming is needed to solve discrete problems.

**Solvers by Objective and Constraint**

| **Constraint Type** | **Objective Type** | | | | |
| --- | --- | --- | --- | --- | --- |
| **Linear** | **Quadratic** | **Least Squares** | **Smooth nonlinear** | **Nonsmooth** |
| None | n/a (*f* = const, or min = ) | [quadprog](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/quadprog.html) | [\](xprod:../../../techdoc/ref/arithmeticoperators.html), [lsqcurvefit](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqcurvefit.html), [lsqnonlin](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqnonlin.html) | [fminsearch](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fminsearch.html), [fminunc](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fminunc.html) | [fminsearch](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fminsearch.html), \* |
| Bound | [linprog](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/linprog.html) | [quadprog](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/quadprog.html) | [lsqcurvefit](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqcurvefit.html), [lsqlin](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqlin.html), [lsqnonlin](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqnonlin.html), [lsqnonneg](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqnonneg.html) | [fminbnd](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fminbnd.html), [fmincon](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fmincon.html), [fseminf](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fseminf.html) | \* |
| Linear | [linprog](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/linprog.html) | [quadprog](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/quadprog.html) | [lsqlin](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/lsqlin.html) | [fmincon](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fmincon.html), [fseminf](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fseminf.html) | \* |
| General smooth | [fmincon](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fmincon.html) | [fmincon](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fmincon.html) | [fmincon](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fmincon.html) | [fmincon](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fmincon.html), [fseminf](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/fseminf.html) | \* |
| Discrete | [bintprog](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/bintprog.html) |  |  |  |  |

The linprog solver solves problems of the form: 

* *fTx* means a row vector of constants *f* multiplying a column vector of variables *x*. In other words,

*fTx* = *f*(1)*x*(1) + *f*(2)*x*(2) + ... + *f*(*n*)*x*(*n*), where *n* is the length of *f*.

* *A · x* ≤ *b* represents linear inequalities. *A* is a *k*-by-*n* matrix, where *k* is the number of inequalities and *n* is the number of variables (size of *x*). *b* is a vector of length *k*. For more information, see [Linear Inequality Constraints](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/brhkghv-11.html#brhkghv-14).
* *Aeq · x* = *beq* represents linear equalities. *Aeq* is an *m*-by-*n* matrix, where *m* is the number of equalities and *n* is the number of variables (size of *x*). *beq* is a vector of length *m*. For more information, see [Linear Equality Constraints](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/brhkghv-11.html#brhkghv-15).
* *lb* ≤ *x* ≤ *ub* means each element in the vector *x* must be greater than the corresponding element of *lb*, and must be larger than the corresponding element of *ub*. For more information, see [Bound Constraints](jar:file:///C:/Program%20Files/MATLAB/R2010b/help/toolbox/optim/help.jar%21/ug/brhkghv-11.html#brhkghv-13).

The syntax of the linprog solver is [x fval] = linprog(f,A,b,Aeq,beq,lb,ub);

**Example 1**

**Suppose a company manufactures two products, A and B, which can be sold for $120 per unit and $80 per unit respectively. Management requires that at least 1000 units be manufactured each month. Product A requires five hours of labor per unit, and product B requires three hours per unit. The cost of labor is $12 per hour, and a total of 8000 hours are available per month. Determine a monthly production schedule that will maximize the company’s profit.**

Here is the set-up to the problem.

Definitions of variables: Let x = the number of units of product A manufactured per month

y = “ “ “ “ “ “ B “ “ “

Objective function: Maximize P (profit) = (120 – 5\*12)x + (80 – 3\*12)y = 60x + 44y

Non-negative constraint: x, y > 0

Problem constraints: x + y > 1000 units manufactured per month

5x + 3y < 8000 labor hours available per month

To solve using the Optimization Toolbox in MATLAB

1) The objective function is to be maximized and the linprog command only works when minimizing the objective function. Therefore you minimize the opposite of the objective function.

Minimize -P = -60x -44y which gives f = [-60 -44]

2) There are two inequalities and both must be of the form ax + by < c. Multiply the first inequality by -1.

-x – y < -1000

5x + 3y < 8000



There are no equations so . The number of columns of *Aeq* should equal the number of variables.

3) Since the variables must be non-negative the lower bound for each is 0. lb = zeros(1,2)

There is no upper bound value for any of the variables so ub doesn’t need to be specified.

MATLAB script

% Example 1 using Optmization Toolbox

% Suppose a company manufactures two products, A and B,

% which can be sold for $120 per unit and $80 per unit respectively.

% Management requires that at least 1000 units be manufactured each month.

% Product A requires five hours of labor per unit,

% and product B requires three hours per unit. The cost of labor is

% $12 per hour, and a total of 8000 hours are available per month.

% Determine a monthly production schedule that will maximize the company’s profit.

% Here is the set-up to the problem.

% Definitions of variables: Let x = the number of units of product A manufactured per month

% y = “ “ “ “ “ “ B “ “ “

% Objective function: Maximize P (profit) = (120 – 5\*12)x + (80 – 3\*12)y = 60x + 44y

% Non-negative constraint: x, y > 0

% Problem constraints: x + y > 1000 units manufactured per month

% 5x + 3y < 8000 labor hours available per month

f = [-60 -44];

A = [-1 -1;5 3]; b = [-1000; 8000];

Aeq = [0 0]; beq = 0;

lb = zeros(1,2);

ub = inf(1,2);

[x fval] = linprog(f,A,b,Aeq,beq,lb)

MATLAB results

Optimization terminated.

x =

1.0e+003 \*

0.0000

2.6667

fval =

-1.1733e+005

Answer: Make 0 product A, 2666.7 product B for a maximum profit of $117330

**Second example**: **A company manufactures chairs, sofas, and loveseats. Each product will need to spend time in the carpentry, finishing, and upholstery divisions. The chart below summarizes the amount of time (in hours) required for each product in each division.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Process**  **Carpentry**  **Finishing**  **Upholstery** | **Chair** | **Sofa** | **Loveseat** |
| **6** | **3** | **2** |
| **1** | **1** | **1** |
| **2** | **6** | **4** |

**Each chair makes a profit of $80, each sofa makes a profit of $70, and each loveseat makes a profit of $66. How many chairs, sofas, and loveseats must be manufactured to maximize profit if there are 96 hours available for carpentry, 18 hours for finishing, and 72 for upholstery?**

Setup: Definitions: x = the # of chairs, y = the # of sofas, z = the # of loveseats.

Maximize P = 80x + 70y + 66z

Subject to: x, y, z, > 0

6x + 3y + 2z < 96 carpentry hours

x + y + z < 18 finishing hours

2x + 6y + 4z < 72 upholstery hours

MATLAB script

% Optimization Example 2

% A company manufactures chairs, sofas, and loveseats.

% Each product will need to spend time in the carpentry,

% finishing, and upholstery divisions. The chart below

% summarizes the amount of time (in hours) required for

% each product in each division.

% Process/Item Chair Sofa Loveseat

% Carpentry 6 3 2

% Finishing 1 1 1

% Upholstery 2 6 4

%Each chair makes a profit of $80, each sofa makes a profit of $70,

% and each loveseat makes a profit of $66. How many chairs, sofas, and

% loveseats must be manufactured to maximize profit if there are 96 hours

% available for carpentry, 18 hours for finishing, and 72 for upholstery?

% Definitions: x = the # of chairs, y = the # of sofas, z = the # of loveseats.

% Maximize P = 80x + 70y + 66z

% Subject to: x, y, z, > 0

% 6x + 3y + 2z < 96 carpentry hours

% x + y + z < 18 finishing hours

% 2x + 6y + 4z < 72 upholstery hours

f = [-80 -70 -66];

A = [6 3 2;1 1 1;2 6 4]; b = [96;18;72];

Aeq = [0 0 0]; beq = 0;

lb = zeros(1,3);

[x fval] = linprog(f,A,b,Aeq,beq,lb)

MATLAB results

Optimization terminated.

x =

14.0000

4.0000

0.0000

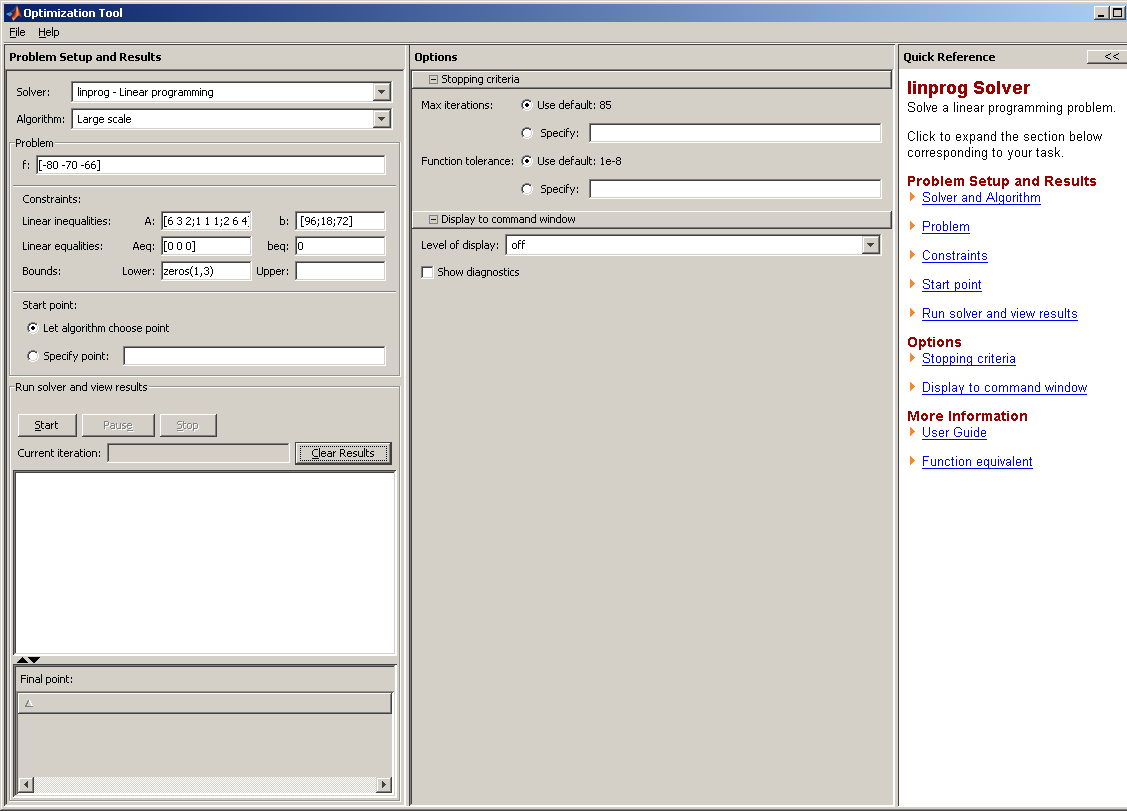
fval =

-1.4000e+003

Answer: Make 14 chairs, 4 sofas and 0 loveseats for a profit of $1400.

**Optimtool**

The command *optimtool* opens up a GUI interface where you can enter the necessary information. Below is example 2.



Once f and the constraints are entered, click the START button. Results are shown below.

