

## LPSolver Extension – v4.2

The LPSolver Extension uses the *lpsolve55* linear programming package (<http://lpsolve.sourceforge.net/5.5/>) to solve linear programming problems, including integer and mixed integer problems. It accepts both maximization and minimization problems. There are therefore two primitives, *lpsolver:max* and *lpsolver:min*. Each is explained below.

### **lpsolver:max numvars constraints objective-function binary-integer**

As suggested by the name, this handles a maximization problem. *lpsolver:max* must be fed the number of variables, the constraints and objective function of the linear programming problem to be solved, and a list indicating whether any of the individual variables are constrained to be integer. Linear programming problems will have the general form:

Maximize

$$c_1X_1 + c_2X_2 + c_3X_3 + \dots$$

Subject to:

$$a_{11}X_1 + a_{12}X_2 + a_{13}X_3 + \dots \leq b_1$$

$$a_{21}X_1 + a_{22}X_2 + a_{23}X_3 + \dots \leq b_2$$

.....

Where the variables ( $X_1, X_2, \dots$ ) are constrained to be non-negative. (The signs on the constraints may be " $\leq$ ", " $\geq$ " or " $=$ ", see below.)

The constraints are provided to NetLogo as a list of coefficient lists:

`[[constraint 1] [constraint 2] [constraint 3] ...]`

where each constraint takes the form

`[a11 a12 .. sign b1]`

To indicate the sign, the following values must be used:

`<= ..... 1`

`>= ..... 2`

`= ..... 3`

The objective function is provided as a single list of coefficients:

[ $c_1 \ c_2 \dots$  ]

One additional list must be provided to designate the individual variables ( $X_1, X_2, \dots$ ) as integer, binary or neither. This takes the form of a simple list, with a numeric value referenced to the variable. The syntax is as follows:

- 0 = no specific constraints
- 1 = variable is an integer variable
- 2 = variable is a binary variable

Therefore, the list [2 1 0] would translate to  $X_1$  binary,  $X_2$  integer,  $X_3$  unconstrained.

Let's look at a simple example.

Maximize

$$3X_1 + 2X_2 + 5X_3$$

subject to:

$$X_1 + 2X_2 + X_3 \leq 430$$

$$3X_1 + 2X_3 \leq 460$$

$$X_1 + 4X_2 \leq 420$$

with neither  $X_1$  nor  $X_2$  nor  $X_3$  being constrained to integer or binary. The NetLogo code might look as follows.

```
let numvar 3
let con [[1 2 1 1 430] [3 0 2 1 460] [1 4 0 1 420]]
let obj [3 2 5]
let bins [0 0 0]
let soln lpsolver:max numvar con obj bins
```

*lpsolver:max* returns a list of solution values. The first item in the list is the value of the objective function, followed by a list of the solution values of variables,  $X_1, X_2$  and  $X_3$ , followed by shadow prices of each constraint. In this case, that would be

[1350.0 [0 100.0 230.0] [1.0 2.0 0]]

Note that with a shadow price of zero, the third constraint is non-binding.

**lpsolver:min numvars constraints objective-function binary-integer**

As suggested by the name, this primitive handles a minimization problem. It is set up in exactly the same way as *lpsolver:max*.

Examples of the use of these primitives are contained in **example\_lpsolver.nlogox**.

LPSolver was written originally by Adam Mackenzie and substantially updated and modified for NetLogo 7.0 by Charles Staelin.

September 2025