Math 5711 Combinatorial optimization, Spring 2004, Vic Reiner
The built-in linear programming functions
in `Maple, Mathematica, MATLAB`

Each of the mathematical packages listed above has built-in functions to solve linear programming functions. I encourage you to try one (or more) of them out, and feel free to use them to *check* your homework solutions.

I'll illustrate the use of each of them on the following problem (in Chvátal's standard form) from lecture:

$$\text{maximize} \qquad 2x_1 + 3x_2$$

$$\begin{array}{lcccl} & x_1 & & & \leq 3 \\ \text{subject to} & x_1 & + & x_2 & \leq 5 \\ & x_1 & + & 2x_2 & \leq 8 \end{array}$$

$$\text{and} \qquad x_1, x_2 \geq 0,$$

which recall had the solution $(x_1, x_2) = (2, 3)$.

Rephrasing the problem in matrix notation:

$$\begin{array}{cc} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ \text{and} & x \geq 0, \end{array}$$

where

$$c = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix} \qquad b = \begin{bmatrix} 3 \\ 5 \\ 8 \end{bmatrix},$$

which has solution

$$x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}.$$

1. Maple

In Maple, there is a package called `simplex` that one must load in, containing various commands implementing Dantzig's simplex method. On my machine, `bonsai.math.umn.edu`, here's how I load it in:

```
bonsai 501 $ maple

    |\^/|     Maple 9 (IBM INTEL LINUX)
._|\|   |/|_. Copyright (c) Maplesoft, a division ....
 \  MAPLE  /  All rights reserved. Maple is a trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
> with(simplex);
Warning, the protected names maximize and
minimize have been redefined and unprotected
[basis, convexhull, cterm, define_zero, display, dual,
feasible, maximize, minimize, pivot, pivoteqn, pivotvar,
ratio, setup, standardize]
```

Once the package is loaded in, you get a help page on the various commands by typing things like `?simplex[maximize];` - note the required semicolon at the end of all `Maple` commands, in order for them to start being processed. In our example, here's what one might do next:

```
> C:=[ x1 <= 3, x1+x2 <= 5, x1+2*x2 <= 8];
            C := [x1 <= 3, x1 + x2 <= 5, x1 + 2 x2 <= 8]

> f:=2*x1+3*x2;
                     f := 2 x1 + 3 x2

> maximize( f, C, NONNEGATIVE );
                    {x2 = 3, x1 = 2}
```

Note the word `NONNEGATIVE` indicating the constraints $x_1, x_2 \geq 0$. At the time of this writing, there was a web page that also explains this:

`www.math.okstate.edu/`
`~wrightd/1493/1493-maple-intro/node4.html`

2. Mathematica

`Mathematica` has a command (`LinearProgramming`) with lots of options:

```
bonsai 518 $ math
Mathematica 5.0 for Linux
Copyright 1988-2003 Wolfram Research, Inc.
 -- Motif graphics initialized --

In[1]:= ?LinearProgramming
LinearProgramming[c, m, b] finds a vector x which
   minimizes the quantity c.x subject to the
   constraints m.x >= b and x >= 0.
LinearProgramming[c, m, {{b1, s1}, {b2, s2}, ... }] finds
   a vector x which minimizes c.x subject to x >= 0 and
   linear constraints specified by the matrix m and the
   pairs {bi, si}. For each row mi of m, the corresponding
   constraint is mi.x >= bi if si == 1, or mi.x == bi if
   si == 0, or mi.x <= bi if si == -1.
LinearProgramming[c, m, b, l] minimizes c.x subject to the
   constraints specified by m and b and x >= l.
LinearProgramming[c, m, b, {l1, l2, ... }] minimizes c.x
   subject to the constraints specified by m and b and
   xi >= li.
LinearProgramming[c, m, b, {{l1, u1}, {l2, u2}, ... }]
   minimizes c.x subject to the constraints specified by
   m and b and li <= xi <= ui.
```

As you can see from the above command summaries, `Mathematica` insists on performing a *minimization* rather than maximization, and prefers the constraint inequalities in the form $Ax \geq b$ rather than Chvátal's $Ax \leq b$. So we should rephrase our problem as minimizing $-c^T x$ subject to $(-A)x \geq (-b)$ with $x \geq 0$, and then we can do it:

```
In[2]:= minusA={{-1,0},{-1,-1},{-1,-2}}

Out[2]= {{-1, 0}, {-1, -1}, {-1, -2}}

In[3]:= minusb={-3,-5,-8}

Out[3]= {-3, -5, -8}
```

```
In[4]:= minusc={-2,-3}

Out[4]= {-2, -3}

In[5]:= LinearProgramming[minusc,minusA, minusb]

Out[5]= {2, 3}
```

3. Matlab

`Matlab` has a command called `linprog`, that is part of its "Optimization toolbox", and can be found described in its help section under that heading. It assumes one wants to solve a *minimization* problem, and the constraints can be of the forms $Ax \leq b$ and/or $Ax = b$ and/or $lb \leq x \leq ub$. Note that it requires one to enter any lower bound (lb) or upper bound constraints (ub) on the variables explicitly.

```
bonsai 590 $ matlab

                          < M A T L A B >
              Copyright 1984-2002 The MathWorks, Inc.
                 Version 6.5.0.180913a Release 13
                           Jun 18 2002

Using Toolbox Path Cache.
Type "help toolbox_path_cache" for more info.

To get started, select "MATLAB Help" from the Help menu.

>> A=[1 0
      1 1
      1 2];
>> b=[3; 5; 8];
>> lb=[ 0; 0];
>> minusc=[-2; -3];
>> linprog(minusc,A,b)
Optimization terminated successfully.

ans =

    2.0000
    3.0000
```