# CS325: Linear programming with Python & Matlab

Prof. Glencora Borradaile

Updated Fall 2015

## The Bicycle Problem

I need to get to Portland as quickly as possible (on my bicycle). The distance is 90 miles but I only have two Burgerville milkshakes (1000 calories and $3 each) to fuel my trip. I can bike 30 miles/hr, but that uses up 17 calories each minute. I could bike really slowly, at 10 miles/hr and only use 3 calories each minute. Or I could split the difference and travel at 20 miles/hr and use 10 calories each minute. What is the fastest way I can reach Portland without running out of energy?

**Objective** Let $t_{30}$,$t_{20}$ and $t_{10}$ be the lengths of time (in minutes) that I will spend travelling 30, 20 and 10 miles/hr. Then, my goal is to:

$$\min t_{30} + t_{20} + t_{10}$$

**Cover enough distance** I need to make sure that those times will allow me to cover the 90 miles to Portland:

$$30 \cdot \frac{t_{30}}{60} + 20 \cdot \frac{t_{20}}{60} + 10 \cdot \frac{t_{10}}{60} = 90$$

Notice that we don't really need an equality, but that a $\geq$ will still allow us to find a feasible solution:

$$30 \cdot \frac{t_{30}}{60} + 20 \cdot \frac{t_{20}}{60} + 10 \cdot \frac{t_{10}}{60} \geq 90$$

Simplifying (this step isn't absolutely necessary):

$$3 \cdot t_{30} + 2 \cdot t_{20} + t_{10} \geq 540$$

**Don't use more energy than we have** I also need to make sure that I don't use up more energy than I have:

$$17 \cdot t_{30} + 10 \cdot t_{20} + 3 \cdot t_{10} \leq 2000$$

**The linear program** Putting it together with the additional observation that our times should not be negative:

$$\begin{aligned} \min \quad & t_{30} + t_{20} + t_{10} \\ s.t. \quad & 3 \cdot t_{30} + 2 \cdot t_{20} + t_{10} \geq 540 \\ & 17 \cdot t_{30} + 10 \cdot t_{20} + 3 \cdot t_{10} \leq 2000 \\ & t_{30} \geq 0 \\ & t_{20} \geq 0 \\ & t_{10} \geq 0 \end{aligned}$$

Now, we just need to find the answer to this problem.

## Using the GLPK LP-solver through Python

We will use the GLPK (GNU Linear Programming Kit) via the PuLP LP modeler for Python to solve this problem.

```
>>> from pulp import *
>>> prob = LpProblem("The Bicycle Problem", LpMinimize)
>>> t10 = LpVariable("t10",0)
>>> t20 = LpVariable("t20",0)
>>> t30 = LpVariable("t30",0)
>>> prob += t10+t20+t30
>>> prob += 3*t30+2*t20+t10 >= 540
>>> prob += 17*t30+10*t20+3*t10 <= 2000
>>> status = prob.solve()
>>> LpStatus[status]
'Optimal'
>>> value(prob.objective)
445.0
>>> value(t10)
397.5
>>> value(t20)
0.0
>>> value(t30)
47.5
```

Conclusion: I can get to Portland in 445 minutes (7 hours and 25 minutes) if I bike at 30 miles per hour for 47.5 minutes and 10 miles per hour for 397.5 minutes, and (if I had to guess) probably in that order. Not too bad for 2 milkshakes. And a lot tastier (and cheaper and more fun) than 5 gallons (or **3 million** calories) of gasoline.

## Using Matlab's linear programming solver

Let's try the same thing again with Matlab's linear programming solver `linprog`. The help page for `linprog` tells us

```
 linprog Linear programming.
    X = linprog(f,A,b) attempts to solve the linear programming problem:

             min f'*x    subject to:   A*x <= b
              x
```

which means that we need to get our LP into the form:

$$\begin{array}{rl} \min\limits_{x} & f'x \\ s.t. & Ax \leq b \end{array}$$

where $x$, $f$ and $b$ are vectors and $A$ is a matrix. Notice that the constraints are only of the form $\leq$. We first need to make that so by negating our $\geq$ constraints:

$$\begin{array}{rr} \min & t_{30} + t_{15} + t_{10} \\ s.t. & -3 \cdot t_{30} - 2 \cdot t_{15} - t_{10} \leq -540 \\ & 17 \cdot t_{30} + 10 \cdot t_{15} + 3 \cdot t_{10} \leq 2000 \\ & -t_{30} \leq 0 \\ & -t_{15} \leq 0 \\ & -t_{10} \leq 0 \end{array}$$

And turning it into matrix form:

$$\min \quad [1\ 1\ 1]\begin{bmatrix} t_{30} \\ t_{15} \\ t_{10} \end{bmatrix}$$

$$s.t. \quad \begin{bmatrix} -3 & -2 & -1 \\ 17 & 10 & 3 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} t_{30} \\ t_{15} \\ t_{10} \end{bmatrix} \leq \begin{bmatrix} -540 \\ 2000 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Putting this into Matlab:

```
>> f = [1;1;1];
>> A = [-3 -2 -1; 17 10 3; -1 0 0; 0 -1 0; 0 0 -1];
>> b = [-540;2000;0;0;0];
>> t = linprog(f,A,b)
Optimization terminated.
t =
   29.5301
   35.9398
  379.5301
>> f'*t
ans = 445.0
```

Conclusion: I can get to Portland in 445 minutes (7 hours and 25 minutes) if I bike at 30 miles per hour for 29.5 minutes, 20 miles per hour for 36 minutes and 10 miles per hour for 379.5 minutes. This is a different answer that gives the same objective. How is it that I can get the same answer two different ways? Watch this video to find out.

## Alternative tools

There are many open-source tools for solving linear programs, and it will not matter which one you use. The wikipedia page on linear programming maintains a list of open-source solves you may opt to use. Matlab is not free, but, while you are a student at OSU, you have access to Matlab through the College of Engineering.

## Python & linear programming on COE computers

You can access the CBC via PuLP and Python quite easily on COE computers via a virtual environment:

1. Set up a virtualenv `http://docs.python-guide.org/en/latest/dev/virtualenvs/`.
2. Install PuLP using easy-install within the virtualenv.

The following should work:

```
% virtualenv venv
New python executable in venv/bin/python
Installing
Setuptools..............................................................................................
Installing
Pip.....................................................................................................
% source venv/bin/activate.csh
[venv] % easy_install -U pulp
```

```
Searching for pulp
...
Finished processing dependencies for pulp
[venv] % python
Python 2.6.6 (r266:84292, Jan 22 2014, 09:42:36)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-4)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from pulp import *
>>>
[venv] % deactivate
%
```