Notes 18 for CS 170

1 Linear Programming

It turns out that a great many problems can be formulated as *linear programs*, i.e., maximizing (or minimizing) a linear function of some variables, subject to *constraints* on the variables; these constraints are either *linear equations* or *linear inequalities*, i.e., linear functions of the variables either set equal to a constant, or $\leq$ a constant, or $\geq$ a constant. Most of this lecture will concentrate on recognizing how to reformulate (or *reduce*) a given problem to a linear program, even though it is not originally given this way. The advantage of this is that there are several good algorithms for solving linear programs that are available. We will only say a few words about these algorithms, and instead concentrate on formulating problems as linear programs.

2 Introductory example in 2D

Suppose that a company produces two products, and wishes to decide the level of production of each product so as to maximize profits. Let x_1 be the amount of Product 1 produced in a month, and x_2 that of Product 2. Each unit of Product 1 brings to the company a profit of 120, and each unit of Product 2 a profit of 500. At this point it seems that the company should only produce Product 2, but there are some constraints on x_1 and x_2 that the company must satisfy (besides the obvious one, $x_1, x_2 \geq 0$). First, x_1 cannot be more than 200, and x_2 more than 300—because of raw material limitations, say. Also, the sum of x_1 and x_2 must be at most 400, because of labor constraints. What are the best levels of production to maximize profits?

We represent the situation by a *linear program*, as follows (where we have numbered the constraints for later reference):

$$\max 120x_1 + 500x_2$$

$$
\begin{array}{ll}
(1) & x_1 \leq 200 \\
(2) & x_2 \leq 300 \\
(3) & x_1 + x_2 \leq 400 \\
(4) & x_1 \geq 0 \\
(5) & x_2 \geq 0
\end{array}
$$

The set of all *feasible* solutions of this linear program (that is, all vectors (x_1, x_2) in 2D space that satisfy all constraints) is precisely the (black) polygon shown in Figure 1 below, with vertices numbered 1 through 5.

The vertices are given in the following table, and labelled in Figure 1 (we explain the meaning of "active constraint" below):

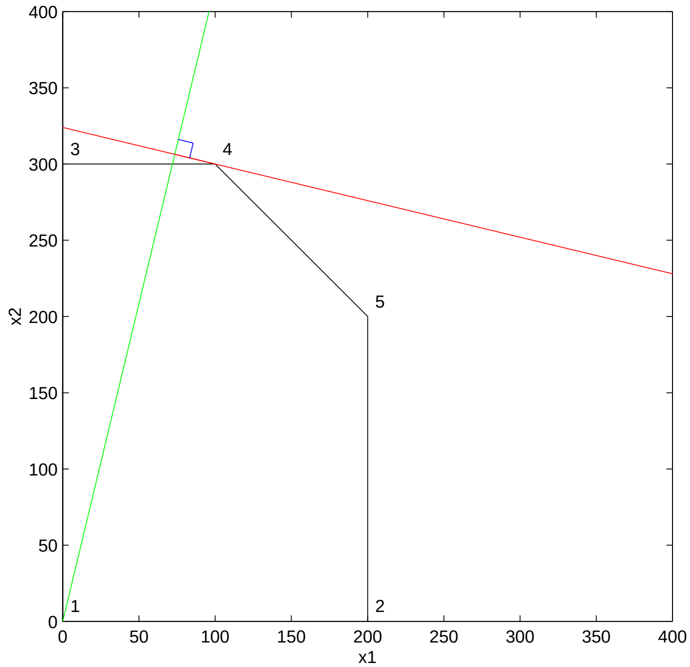

Figure 1: The feasible region (polygon), solution vertex (#4), and line of constant profit

Vertex	x1-coord	x2-coord	Active constraints
1	0	0	4,5
2	200	0	1,5
3	0	300	2,4
4	100	300	2,3
5	200	200	1,3

The reason all these constraints yield the polygon shown is as follows. Recall that a linear equation like $ax_1 + bx_2 = p$ defines a line in the plane. The *inequality* $ax_1 + bx_2 \leq p$ defines all points on one side of that line, i.e., a *half-plane*, which you can think of as an (infinite) polygon with just one side. If we have two such constraints, the points have to lie in the intersection of two half-planes, i.e., a polygon with 2 sides. Each constraint adds (at most) one more side to the polygon. For example, the 5 constraints above yield 5 sides in the polyhedron: constraint (1), $x_2 \leq 200$, yields the side with vertices #2 and #5, constraint (2), $x_3 \leq 300$, yields the side with vertices #3 and #4, constraint (3), $x_1 + x_2 \leq 400$, yields the side with vertices #4 and #5, constraint (4), $x_1 \geq 0$, yields the side with vertices #1 and #3, and constraint (5), $x_2 \geq 0$, yields the side with vertices #1 and #2. We also say that constraint (1) is *active* at vertices #2 and #5 since it is just barely satisfied at those vertices (at other vertices x_2 is strictly less than 200).

We wish to maximize the linear function profit$=120x_1 + 500x_2$ over all points of this polygon. We think of this geometrically as follows. The set of all points satisfying $p = 120x_1 + 500x_2$ for a fixed p is a line. As we vary p, we get different lines, all parallel to one another, and all perpendicular to the vector $(120, 500)$. (We review this basic geometrical fact below).

Geometrically, we want to increase p so that the line is just barely touching the polygon at one point, and increasing p would make the plane miss the polygon entirely. It should be clear geometrically that this point will usually be a *vertex* of the polygon. This point is the *optimal solution* of the linear program. This is shown in the figure above, where the green line (going from the origin to the top of the graph) is parallel to the vector $(120, 500)$, the red line (going all the way from left to right across the graph) is perpendicular to the green line and connects to the solution vertex #4 $(100, 300)$, which occurs for $p = 120 * 100 + 500 * 300 = 162000$ in profit. (The blue "L" connecting the green and red lines indicates that they are perpendicular.)

(Now we review why the equation $y_1 \cdot x_1 + y_2 \cdot x_2 = p$ defines a line perpendicular to the vector $y = (y_1, y_2)$. You may skip this if this is familiar material. Write the equation as a dot product of y and $x = (x_1, x_2)$: $y * x = p$. First consider the case $p = 0$, so $y * x = 0$. Recall that the if the dot product of two vectors is 0, then the vectors are perpendicular. So when $p = 0$, $y * x = 0$ defines the set of all vectors (points) x perpendicular to y, which is a line through the origin. When $p \neq 0$, we argue as follows. Note that $y * y = y_1^2 + y_2^2$. Then define the vector $\bar{y} = (p/(y * y))y$, a multiple of y. Then we can easily confirm that $\bar{y}$ satisfies the equation because $y * \bar{y} = (p/(y * y))(y * y) = p$. Now think of every point x as the sum of two vectors $x = \bar{x} + \bar{y}$. Substituting in the equation for x we get $p = y * x = y * (\bar{x} + \bar{y}) = y * \bar{x} + y * \bar{y} = y * \bar{x} + p$, or $y * \bar{x} = 0$. In other words, the points $\bar{x}$ lie in a plane through the origin perpendicular to y, and the points $x = \bar{x} + \bar{y}$ are gotten just by adding the vector $\bar{y}$ to each vector in this plane. This just shifts the plane in the direction $\bar{y}$, but leaves it perpendicular to y.)

There are three other geometric possibilities that could occur:

- If the planes for each p are parallel to an edge or face touching the solution vertex, then all points in that edge or face will also be solutions. This just means that the solution is not unique, but we can still solve the linear program. This would occur in the above example if we changed the profits from $(120, 500)$ to $(100, 100)$; we would get equally large profits of $p = 40000$ either at vertex #5 $(200, 200)$, vertex #4 $(100, 300)$, or anywhere on the edge between them.

- It may be that the polygon is *infinite*, and that p can be made arbitrarily large. For example, removing the constraints $x_1 + x_2 \leq 400$ and $x_1 \leq 200$ means that x_1 could become arbitrarily large. Thus $(x_1, 0)$ is in the polygon for all $x_1 > 0$, yielding an arbitrarily large profit $120x_1$. If this happens, it probably means you forgot a constraint and so formulated your linear program incorrectly.

- It may be that the polygon is *empty*, which is also called *infeasible*. This means that *no* points (x_1, x_2) satisfy the constraints. This would be the case if we added the constraint, say, that $x_1 + 2x_2 \geq 800$; since the largest value of $x_1 + 2x_2$ occurs at vertex #4, with $x_1 + 2x_2 = 100 + 2 * 300 = 700$, this extra constraint cannot be satisfied. When this happens it means that your problem is overconstrained, and you have to weaken or eliminate one or more constraints.

3 Introductory Example in 3D

Now we take the same company as in the last section, add Product 3 to its product line, along with some constraints, and ask how the problem changes. Each unit of Product 3 brings a profit of 200, and the sum of x_2 and three times x_3 must be at most 600, because Products 2 and 3 share the same piece of equipment ($x_2 + 3x_3 \leq 600$).

This changes the linear program to

$$
\begin{array}{lll}
\max 120x_1 + 500x_2 + 200x_3 & & \\
& (1) & x_1 \leq 200 \\
& (2) & x_2 \leq 300 \\
& (3) & x_1 + x_2 \leq 400 \\
& (4) & x_1 \geq 0 \\
& (5) & x_2 \geq 0 \\
& (6) & x_3 \geq 0 \\
& (7) & x_2 + 3x_3 \leq 600
\end{array}
$$

Each constraint correspond to being on one side of a plane in (x_1, x_2, x_3) space, a *half-space*. The 7 constraints result in a 7-sided polyhedron shown in Figure 2. The polyhedron has vertices and active constraints show here:

Vertex	x1-coord	x2-coord	x3-coord	Active constraints
1	0	0	0	4,5,6
2	200	0	0	1,5,6
3	0	300	0	2,4,6
4	100	300	0	2,3,6
5	200	200	0	1,3,6
6	0	0	200	4,5,7
7	100	300	100	2,3,7
8	200	0	200	1,5,7

Note that a vertex now has 3 active constraints, because it takes the intersection of at least 3 planes to make a corner in 3D, whereas it only took the intersection of 2 lines to make a corner in 2D.

Again the (green) line is in the direction (120,500,200) of increasing profit, the maximum of which occurs at vertex #7. There is a (red) line connecting vertex #7 to the green line, to which it is perpendicular.

In general m constraints on n variables can yield an m-sided polyhedron in n-dimensional space. Such a polyhedron can be seen to have as many as $\binom{m}{n}$ vertices, since n constraints are active at a corner, and there are $\binom{m}{n}$ ways to choose n constraints. Each of these very many vertices is a candidate solution. So when m and n are large, we must rely on a systematic algorithm rather than geometric intuition in order to find the solution.

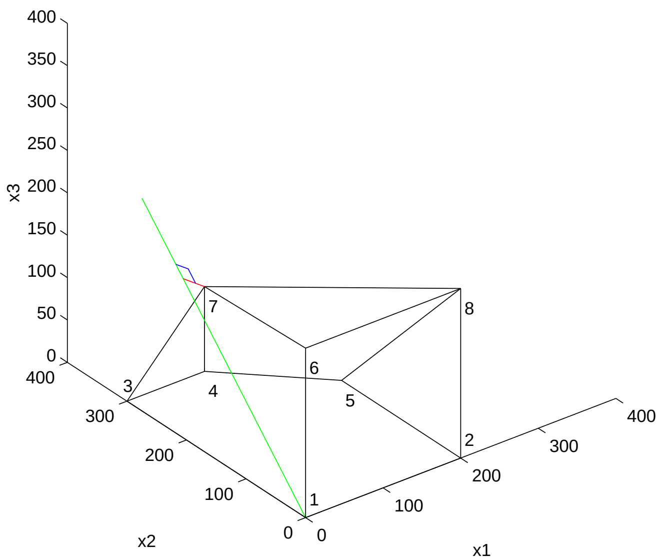

Figure 2: The feasible region (polyhedron).

4 Algorithms for Linear Programming

Linear programming was first solved by the *simplex method* devised by George Dantzig in 1947.

We describe the algorithm with reference to last lecture's example. The linear program was

$$\max 120x_1 + 500x_2 + 200x_3$$

$$
\begin{aligned}
&(1) \quad & x_1 &\leq 200 \\
&(2) \quad & x_2 &\leq 300 \\
&(3) \quad & x_1 + x_2 &\leq 400 \\
&(4) \quad & x_1 &\geq 0 \\
&(5) \quad & x_2 &\geq 0 \\
&(6) \quad & x_3 &\geq 0 \\
&(7) \quad & x_2 + 3x_3 &\leq 600
\end{aligned}
$$

and the polyhedron of feasible solutions is shown in Figure 3 and its set of vertices is

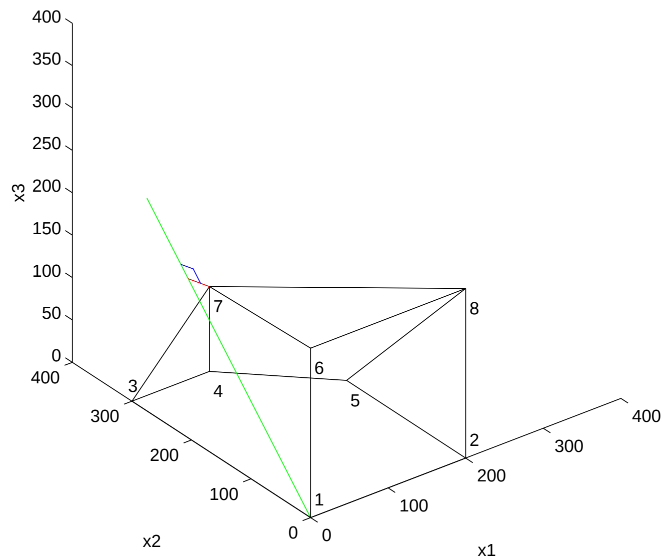

Figure 3: The feasible region (polyhedron).

Vertex	x1-coord	x2-coord	x3-coord	Active constraints
1	0	0	0	4,5,6
2	200	0	0	1,5,6
3	0	300	0	2,4,6
4	100	300	0	2,3,6
5	200	200	0	1,3,6
6	0	0	200	4,5,7
7	100	300	100	2,3,7
8	200	0	200	1,5,7

The simplex method starts from a vertex (in this case the vertex $(0,0,0)$) and repeatedly looks for a vertex that is adjacent, and has better objective value. That is, it is a kind of *hill-climbing* in the vertices of the polytope. When a vertex is found that has no better neighbor, simplex stops and declares this vertex to be the optimum. For example, in Figure 2, if we start at vertex #1 $(0,0,0)$, then the adjacent vertices are #2, #3, and #4 with profits 24000, 150000 and 40000, respectively. If the algorithm chooses to go to #3, it then examines vertices #6 and #7, and discovers the optimum #7. There are now implementations of simplex that solve routinely linear programs with *many* thousands of variables and constraints.

The simplex algorithm will also discover and report the other two possibilities: that the solution is infinite, or that the polyhedron is empty. In the worst case, the simplex algorithm takes exponential time in n, but this is very rare, so simplex is widely used in practice. There are other algorithms (by Khachian in 1979 and Karmarkar in 1984) that are guaranteed to run in polynomial time, and are sometimes faster in practice.

5 Different Ways to Formulate a Linear Programming Problem

There are a number of equivalent ways to write down the constraints in a linear programming problem. Some formulations of the simplex method use one and some use another, so it is important to see how to transform among them.

One standard formulation of the simplex method is with a matrix A of constraint coefficients, a vector b of constraints, and a vector f defining the linear function $f * x = \sum_i f_i \cdot x_i$ (the dot product of f and x) to be maximized. The constraints are written as the single inequality $A \cdot x \leq b$, which means that every component $(A \cdot x)_i$ of the vector $A \cdot x$ is less than or equal to the corresponding component b_i: $(A \cdot x)_i \leq b_i$. Thus, A has as many rows as there are constraints, and as many columns as there are variables.

In the example above, $f = [120, 500, 200]$,

$$A = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 3 \end{matrix} \quad \text{and} \quad b = \begin{matrix} 200 \\ 300 \\ 400 \\ 0 \\ 0 \\ 0 \\ 600 \end{matrix}$$

Note that the constraints 4, 5 and 6 are $-x_1 \leq 0$, $-x_2 \leq 0$ and $-x_3 \leq 0$, or $x_1 \geq 0$, $x_2 \geq 0$ and $x_3 \geq 0$, respectively. In other words, constraints with $\geq$ can be changed into $\leq$ just by multiplying by -1.

Note that by changing f to $-f$, and maximizing $-f * x$, we are actually *minimizing* $f * x$. So linear programming handles both maximization and minimization equally easily.

Matlab 5.3, which is available on UNIX machines across campus, has a function linprog$(-f, A, b)$ for solving linear programs in this format. (This implementation *minimizes* the linear function instead of maximizing it, but since minimizing $-f * x$ is the same as maximizing $f * x$, we only have to negate the f input argument to get Matlab to maximize $f * x$). In earlier Matlab versions this program is called LP.

A Java applet with a nice GUI for solving linear programming problems is available at URL riot.ieor.berkeley.edu/riot (click on "Linear Program Solver with Simplex").

Now suppose that in addition to inequalities, we have equalities, such as $x_1 + x_3 = 10$. How do we express this in terms of inequalities? This is simple: write each equality as *two* inequalities: $x_1 + x_3 \leq 10$ and $x_1 + x_3 \geq 10$ (or $-x_1 - x_2 \leq -10$).

Similarly, one can turn any linear program into one just with equalities, and all inequalities of the form $x_i \geq 0$; some versions of simplex require this form. To turn an inequality like $x_1 + x_2 \leq 400$ into an equation, we introduce a new variable s (the *slack variable* for this inequality), and rewrite this inequality as $x_1 + x_2 + s = 400, s \geq 0$. Similarly, any inequality like $x_1 + x_3 \geq 20$ is rewritten as $x_1 + x_3 - s = 20, s \geq 0$; s is now called a *surplus* variable.

We handle an unrestricted variable x as follows: We introduce two nonnegative variables, x^+ and x^-, and replace x by $x^+ - x^-$. This way, x can take on any value.

6 A Production Scheduling Example

We have the demand estimates for our product for all months of 1997, $d_i : i = 1, \ldots, 12$, and they are very uneven, ranging from 440 to 920. We currently have 60 employees, each of which produce 20 units of the product each month at a salary of 2,000; we have no stock of the product. How can we handle such fluctuations in demand? Three ways:

- overtime—but this is expensive since it costs 80% more than regular production, and has limitations, as workers can only work 30% overtime.
- hire and fire workers—but hiring costs 320, and firing costs 400.
- store the surplus production—but this costs 8 per item per month

This rather involved problem can be formulated and solved as a linear program. As in all such reductions, a crucial first step is defining the variables:

- Let w_i be the number of workers we have in the ith month—we start with $w_0 = 60$.
- Let x_i be the production for month i.
- o_i is the number of items produced by overtime in month i.
- h_i and f_i are the numbers of workers hired/fired in the beginning of month i.
- s_i is the amount of product stored after the end of month i.

We now must write the constraints:

- $x_i = 20w_i + o_i$—the amount produced is the one produced by regular production, plus overtime.
- $w_i = w_{i-1} + h_i - f_i, w_i \geq 0$—the changing number of workers.
- $s_i = s_{i-1} + x_i - d_i \geq 0$—the amount stored in the end of this month is what we started with, plus the production, minus the demand.
- $o_i \leq 6w_i$—only 30% overtime.

Finally, what is the objective function? It is

$$\min 2000 \sum w_i + 400 \sum f_i + 320 \sum h_i + 8 \sum s_i + 180 \sum o_i.$$

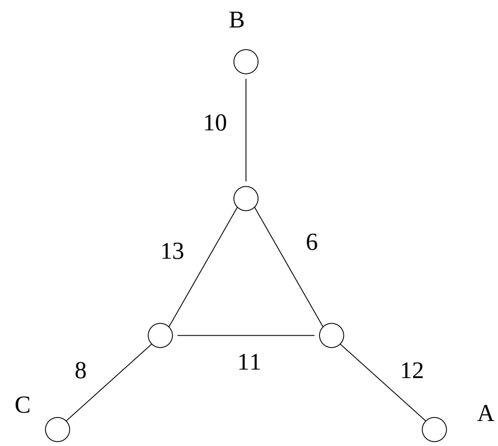

Figure 4: A communication network

7 A Communication Network Problem

We have a network whose lines have the bandwidth shown in Figure 4. We wish to establish three calls: One between A and B (call 1), one between B and C (call 2), and one between A and C (call 3). We must give each call at least 2 units of bandwidth, but possibly more. The link from A to B pays 3 per unit of bandwidth, from B to C pays 2, and from A to C pays 4. Notice that each call can be routed in two ways (the long and the short path), or by a combination (for example, two units of bandwidth via the short route, and three via the long route). How do we route these calls to maximize the network's income?

This is also a linear program. We have variables for each call and each path (long or short); for example x_1 is the short path for call 1, and x'_2 the long path for call 2. We demand that (1) no edge bandwidth is exceeded, and (2) each call gets a bandwidth of 2.

$$
\begin{aligned}
\max 3x_1 + 3x'_1 + 2x_2 + 2x'_2 + 4x_3 + 4x'_3 & \\
x_1 + x'_1 + x_2 + x'_2 &\leq 10 \\
x_1 + x'_1 + x_3 + x'_3 &\leq 12 \\
x_2 + x'_2 + x_3 + x'_3 &\leq 8 \\
x_1 + x'_2 + x'_3 &\leq 6 \\
x'_1 + x_2 + x'_3 &\leq 13 \\
x'_1 + x'_2 + x_3 &\leq 11 \\
x_1 + x'_1 &\geq 2 \\
x_2 + x'_2 &\geq 2 \\
x_3 + x'_3 &\geq 2 \\
x_1, x'_1 \ldots, x'_3 &\geq 0
\end{aligned}
$$

The solution, obtained via simplex in a few milliseconds, is the following: $x_1 = 0, x'_1 = 7, x_2 = x'_2 = 1.5, x_3 = .5, x'_3 = 4.5$.

8 The Simplex Algorithm

Consider the following Linear Programming (LP) problem:

$$
\begin{array}{lll}
\text{maximize } z & = & 5x_1 + 5x_2 + 3x_3 \\
\text{subject to} & & \begin{array}{lrrrcl}
(1) & x_1 & +3x_2 & +x_3 & \leq & 3 \\
(2) & -x_1 & & +3x_3 & \leq & 2 \\
(3) & 2x_1 & -x_2 & +2x_3 & \leq & 4 \\
(4) & 2x_1 & +3x_2 & -x_3 & \leq & 2
\end{array} \\
\text{where } x_1, x_2, x_3 \geq 0 & &
\end{array}
$$

The inequalities (1)–(4) and $x_1, x_2, x_3 \geq 0$ define 7 half-spaces in R^3. The intersection of these half-spaces is the feasible region. The corresponding equations define 7 planes in R^3. Any 3 of these planes intersect in a single point; there are $\binom{7}{3} = 35$ such points. Not all of these points are corner vertices of the feasible region: the point $(x_1, x_2, x_3) = (0, 0, 0)$ is a corner verex (it satisfies inequalities (1)–(4)); the point $(3, 0, 0)$, which lies at the intersection of the equation corresponding to (1) and $x_2 = 0$ and $x_3 = 0$, is not (it violates inequalities (3) and (4)).

We introduce slack variables:

$$
\begin{array}{lll}
\text{maximize } z & = & 5x_1 + 5x_2 + 3x_3 \\
\text{subject to} & & \begin{array}{lrrrrrrrcl}
(1') & x_1 & +3x_2 & +x_3 & +x_4 & & & & = & 3 \\
(2') & -x_1 & & +3x_3 & & +x_5 & & & = & 2 \\
(3') & 2x_1 & -x_2 & +2x_3 & & & +x_6 & & = & 4 \\
(4') & 2x_1 & +3x_2 & -x_3 & & & & +x_7 & = & 2
\end{array} \\
\text{where } x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geq 0 & &
\end{array}
$$

This gives us 4 equations in 7 variables. If we set 3 of the variables to 0, then we have 4 equations in 4 variables, which gives us a unique solution (provided the equations are linearly independent, which they are in this example). These $\binom{7}{3} = 35$ solutions define the points at the intersections of 3 planes from the original formulation. Moreover, the solutions that have no negative components define exactly the corner vertices. For example, if we set $x_2 = x_3 = x_4 = 0$, then the unique solution is $(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (3, 0, 0, 0, 5, -2, -4)$, which is not feasible ($x_6 \geq 0$ and $x_7 \geq 0$ are violated). The solution obtained from setting the non-slack variables to 0, which defines the corner vertex $(0, 0, 0)$, is always feasible: $(0, 0, 0, 3, 2, 4, 2)$. For this corner vertex the objective function is $z = 0$.

We want to do better by moving to a neighboring corner vertex. This corresponds to removing one of the 3 planes $x_1, x_2, x_3 = 0$ and replacing it with another plane. Suppose we try to increase the value of x_1, keeping $x_2 = x_3 = 0$. How much can we increase x_1? Equation (1′) tells us that $x_1 \leq 3$ (because $x_4 \geq 0$). Similarly, equation (3′) implies $x_1 \leq 2$, and equation (4′) implies $x_1 \leq 1$ (note that equation (1′) provides no upper bound on x_1). The sharpest of these constraints is $x_1 \leq 1$, which implies $x_7 = 0$. If we set $x_1 = 1$ (and $x_2 = x_3 = 0$), then the unique solution is $(1, 0, 0, 2, 3, 2, 0)$, which defines the corner vertex $(1, 0, 0)$. Note that this vertex lies at the intersection of the equation corresponding to (4) and $x_2 = 0$ and $x_3 = 0$, that is, we replaced the plane $x_1 = 0$ by (4). At vertex $(1, 0, 0)$, the objective function is $z = 5$, which is an improvement over $z = 0$. Can we do better by moving to a neighboring vertex?

In the previous improvement, we changed the *basis* (the non-zero variables) from x_4, x_5, x_6, x_7 to x_1, x_4, x_5, x_6. This was easy to do as the coefficients of the original basis x_4, x_5, x_6, x_7 in equations $(1')$–$(4')$ were the 4×4 identity matrix. In order to perform a second basis switch, we first want to convert equations $(1')$–$(4')$ into equivalent equations (i.e., equations having the same solutions) such that the coefficients of the new basis x_1, x_4, x_5, x_6 are the identity matrix. Equivalence is preserved if we add a linear combination of the equations to any one equation (why?). First, in order to obtain coefficient 1 of x_1 in equation $(4')$, we divide equation $(4')$ by 2. Second, in order to obtain coefficient 0 of x_1 in equation $(1')$, we subtract the new equation $(4'')$ from $(1')$. Third, in order to obtain coefficient 0 of x_1 in equation $(2')$, we add the new equation $(4'')$ to $(2')$. Fourth, in order to obtain coefficient 0 of x_1 in equation $(3')$, we subtract the new equation $(4'')$ twice from $(3')$. The resulting LP problem is equivalent to the original one:

$$
\begin{array}{lllllllllll}
\text{maximize } z & = & \multicolumn{9}{l}{5x_1 + 5x_2 + 3x_3} \\
 & (1'') & & \frac{3}{2}x_2 & +\frac{3}{2}x_3 & +x_4 & & & -\frac{1}{2}x_7 & = & 2 \\
\text{subject to} & (2'') & & \frac{3}{2}x_2 & +\frac{5}{2}x_3 & & +x_5 & & +\frac{1}{2}x_7 & = & 3 \\
 & (3'') & & -4x_2 & +3x_3 & & & +x_6 & -x_7 & = & 2 \\
 & (4'') & x_1 & +\frac{3}{2}x_2 & -\frac{1}{2}x_3 & & & & +\frac{1}{2}x_7 & = & 1 \\
\multicolumn{11}{l}{\text{where } x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geq 0}
\end{array}
$$

Now suppose we want to bring x_3 into the basis, that is, we want to increase the value of x_3 as much as possible subject to equations $(1'')$–$(4'')$. According to $(1'')$, we have $x_3 \leq 4/3$; according to $(2'')$, we have $x_3 \leq 6/5$; and according to $(3'')$, we have $x_3 \leq 2/3$. The last is the sharpest constraint, and therefore bringing x_3 into the basis means removing x_6. The new solution, at basis x_1, x_3, x_4, x_5, is $(\frac{4}{3}, 0, \frac{2}{3}, 1, \frac{4}{3}, 0, 0)$. This gives the value $z = \frac{26}{3}$ of the objective function, and the following new LP problem:

$$
\begin{array}{lllllllllll}
\text{maximize } z & = & \multicolumn{9}{l}{5x_1 + 5x_2 + 3x_3} \\
 & (1''') & & \frac{7}{2}x_2 & & +x_4 & & -\frac{1}{2}x_6 & & = & 1 \\
\text{subject to} & (2''') & & \frac{29}{6}x_2 & & & +x_5 & -\frac{5}{6}x_6 & +\frac{4}{3}x_7 & = & \frac{4}{3} \\
 & (3''') & & -\frac{4}{3}x_2 & +x_3 & & & +\frac{1}{3}x_6 & -\frac{1}{3}x_7 & = & \frac{2}{3} \\
 & (4''') & x_1 & +\frac{5}{6}x_2 & & & & +\frac{1}{6}x_6 & +\frac{1}{3}x_7 & = & \frac{4}{3} \\
\multicolumn{11}{l}{\text{where } x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geq 0}
\end{array}
$$

Let us back up a bit. How did we decide to bring x_3 into the basis and not, say, x_2? We could have done a calculation and seen that with x_2, the sharpest constraint would have been given by equation $(4'')$, namely, $x_2 \leq 2/3$. Moreover, from equation $(4')$ we see that adding $2/3$ units of x_2 would remove x_1 from the basis, and thus cause a net *decrease* in the objective function of $-5 \cdot 1 + 5 \cdot \frac{2}{3} = -\frac{5}{3}$. We can speed up this computation by also adjusting the basis of the objective function, that is, by representing z not by the initial variables x_1, x_2, x_3, but by the current non-basis variables. Initially, with basis x_4, x_5, x_6, x_7 (the slack variables), we have:

$$z = 0 + 5x_1 + 5x_2 + 3x_3.$$

With the first *pivot* step, we remove x_1 from the objective function by subtracting 5 times equation (4″):

$$z \;=\; 5 - \frac{5}{2}x_2 + \frac{11}{2}x_3 - \frac{5}{2}x_7.$$

Note that this representation of the objective function gives us its current value (namely, 5), and how the introduction of one unit of x_2, x_3, or x_7 into the basis would change that value. Only the introduction of x_3 increases the objective function, so this is our choice. With the second pivot step, which replaces x_6 in the basis by x_3, we obtain the following *tableau*:

$$\begin{array}{lllllllllll}
\text{maximize } z \;=\; \frac{26}{3} + \frac{29}{6}x_2 - \frac{11}{6}x_6 - \frac{2}{3}x_7 \\
 & (1''') & & \frac{7}{2}x_2 & & +x_4 & & -\frac{1}{2}x_6 & & = & 1 \\
\text{subject to} & (2''') & & \frac{29}{6}x_2 & & & +x_5 & -\frac{5}{6}x_6 & +\frac{4}{3}x_7 & = & \frac{4}{3} \\
 & (3''') & & -\frac{4}{3}x_2 & +x_3 & & & +\frac{1}{3}x_6 & -\frac{1}{3}x_7 & = & \frac{2}{3} \\
 & (4''') & x_1 & +\frac{5}{6}x_2 & & & & +\frac{1}{6}x_6 & +\frac{1}{3}x_7 & = & \frac{4}{3} \\
\text{where } x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geq 0
\end{array}$$

This gives us the following information: the current solution is $(\frac{4}{3}, 0, \frac{2}{3}, 1, \frac{4}{3}, 0, 0)$ (see right-hand sides of the equations), the current corner vertex is $(x_1, x_2, x_3) = (\frac{4}{3}, 0, \frac{2}{3})$ (remove slack variables), the current value of the objective function is $\frac{26}{3}$, and this value can be improved by introducing x_2 into the basis.

In the next (third) pivot step, we bring x_2 into the basis and remove x_5. The new tableau is:

$$\begin{array}{llllllllll}
\text{maximize } z \;=\; 10 - x_5 - x_6 - 2x_7 \\
 & (1'''') & & & & x_4 & -\frac{21}{29}x_5 & +\frac{3}{29}x_6 & -\frac{28}{29}x_7 & = \; \frac{1}{29} \\
\text{subject to} & (2'''') & & x_2 & & & \frac{6}{29}x_5 & -\frac{5}{29}x_6 & +\frac{8}{29}x_7 & = \; \frac{8}{29} \\
 & (3'''') & & & x_3 & & \frac{8}{29}x_5 & +\frac{3}{29}x_6 & +\frac{1}{29}x_7 & = \; \frac{30}{29} \\
 & (4'''') & x_1 & & & & -\frac{5}{29}x_5 & +\frac{9}{29}x_6 & +\frac{3}{29}x_7 & = \; \frac{32}{29} \\
\text{where } x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geq 0
\end{array}$$

The new solution is $(\frac{32}{29}, \frac{8}{29}, \frac{30}{29}, \frac{1}{29}, 0, 0, 0)$. At the corresponding vertex $(x_1, x_2, x_3) = (\frac{32}{29}, \frac{8}{29}, \frac{30}{29})$, the objective function is $z = 10$. This is the optimum, because any change in the basis would decrease it. (This is because for LP problems, due to the convexity of the feasible region, it can be shown that if we are at a corner vertex v such that all neighboring vertices of v have no better value of the objective function, then v is indeed a global optimum.)

In each pivot step, the Simplex algorithm moves from one corner vertex to a neighboring vertex with a better value of the objective function, by replacing one variable in the basis. There may be several choices for new basis variables that would increase the objective function. If this is the case, then any choice means progress, and there are several heuristics trying to achieve maximal progress. Once a new basis variable has been chosen, there are three possible outcomes. First (as in the above example), some equation may provide the sharpest upper-bound constraint on the new basis variable, and this determines which current basis variable is removed. Second, no equation may provide an upper bound on the new basis variable. In this case, the feasible region is unbounded, and the value of the objective function can be made arbitrarily large by introducing the new basis variable.

Third, there may be ties in case several equations provide the same upper bound on the new basis variable. This case is problematic, and one has to carefully (e.g., randomly) choose which basis variable to replace in order to avoid "cycling," i.e., moving back and forth between the same corner vertices without making progress. We will not discuss this further.

9 Duality

Every LP problem has a *dual* formulation, which is obtained by transposing the coefficient matrix and has the same optimum. For the maximization problem from the previous section, by duality we get the following minimization problem:

$$
\begin{array}{lllllll}
\text{minimize } z & = & \multicolumn{5}{l}{3y_1 + 2y_2 + 4y_3 + 2y_4} \\
 & \text{(A)} & y_1 & -y_2 & +2y_3 & +2y_4 & \geq 5 \\
\text{subject to} & \text{(B)} & 3y_1 & & -y_3 & +3y_4 & \geq 5 \\
 & \text{(C)} & y_3 & +3y_2 & +2y_3 & -y_4 & \geq 3 \\
\text{where } y_1, y_2, y_3, y_4 \geq 0 & & & & & &
\end{array}
$$

Suppose this has the following interpretation: there are 4 foods and three 3 nutrients. Each variable y_i represents the number of annually consumed units of food i. The cost per unit of food 1 is 3, one unit of food 2 costs 2, etc. Equation (A) specifies that consuming 1 unit of food 1 adds 1 unit of nutrient 1 to the diet, but consuming 1 unit of food 2 burns one unit of nutrient 1. Similarly, each unit of food 3 and food 4 add two units of nutrient 1. The total yearly requirement for nutrient 1 is 5 units. Equation (B) refers similarly to nutrient 2, and equation (C) to nutrient 3. Then the solution asks for an annual diet that satisfies all nutritional requirements at the least cost.

Now recall the dual version of the problem from the previous section. What is its interpretation? Each variable x_j can be thought of representing the price a pill maker charges per unit of nutrient j. The constraints (1)–(4) specify that the pill makes needs to be competitive with the price of real food: 1 unit of food 1 costs 3, so pills with equivalent nutrient contents must not cost more than 3, etc. The objective function maximizes the profit of the pill maker, assuming that the customer buys as many pills as needed to satisfy his nutritional requirements. By duality, the minimal cost of the consumer's diet is equal to the maximal profit of the pill maker.