LINEAR PROGRAMMING

JENNIFER REN, SAEE

ABSTRACT. We will first introduce some of the classical linear programming problems, and then go over definitions and properties that are important such as duality. These problems focus mainly on optimization and we will go over why linear programming algorithms will be more efficient than other algorithms. Then we will introduce algorithms and utilize them to solve the problems we introduced. We finally end with comparing the efficiencies of each algorithm and the pros and cons.

1. Introducing the Problem

A *linear program* consists of a set of constraints on n variables and an objective function which needs to be optimized.

Definition 1.1 (a standard linear program)**.** maximize $c_1x_1 + c_2x_2 + \cdots + c_nx_n$ subject to:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq b_m$$

$$x_1, x_2, \cdots, x_n \geq 0$$

Notation. The standard LP can be written compactly in terms of vectors and matrices. We will use the same notation throughout.

$$\max\ c^T\mathbf{x}$$

$$A\mathbf{x} \leq b$$

$$\mathbf{x} \geq 0$$

Definition 1.2 (integer program)**.** An integer program is a linear program where we look for integral solutions.

We are often looking for integral solutions, and thus defining an integer program is the next natural thing to do. It can be shown that an integer program has no deterministic polynomial time algorithm (consider the example below). However, we can use a linear program to get decent approximations (by using various rounding methods) for the same.

Date: March 20, 2023.

1.1. **Some History.** Such a setup was often used to solve optimization problems like transporting goods, scheduling, etc in the World War II. A prelimanry observation is that a linear program can be solved by using brute force in $O(n^m)$ time - which is clearly not good. The first feasible algorithm for this was devised by George B. Dantzig in the 1940s, known as the Simplex Algorithm. As we see in this paper, this method simplified the problem a lot. However, the runtime of the algorithm is exponential in the cases when the numer of variables/constraints is huge. Leonid Khachiyan, in 1979 gave a method called the Ellipsoid Method which runs in polynomial time in the number of variables. However, it appears to be doing worse in implementation that the Simplex Method. In 1984, Narendra Karmakar discovered yet another polynomial time algorithm (also mentioned later in this paper) which was at par with the Simplex Method (knows as the Interior Point Method).

1.2. **Examples.** Before we get into the algorithms, let us look at some problems in theoretical computer science which may not seem to be in the above form, but can be approximately (and sometimes efficiently) reduced to a Linear Program.

Example (Vertex Cover). This NP-complete problem takes a graph $G = (V, E)$ with vertex i having weight w_i as input (can be thought of as unweighted by setting every vertex to weight 1) and must return a set $S \subseteq V$ such that for all edges $e = (u, v) \in E$, at least one of u and v belongs to S and $\sum_{i:v_i \in S} w_i$ is minimized.
A corresponding Integer Program can be formulated as follows: let x_i be a variable corresponding to vertex i. We let it be between 0 and 1. Note that since this is an Integer Program, x_i is either 0 or 1. It being 0 corresponds to the vertex not belonging to the cover, and we include the vertex if the value is 1. Further, the constraint of every edge being covered can be written as follows:

$$x_u + x_v \geq 1 \, \forall (u, v) \in E.$$

Similarly, the objective function that we want to minimize is

$$\sum_{i=1}^{|V|-n} w_i x_i$$

It can be shown that rounding of the optimal solution obtained by solving this as a *linear* program (instead of integer) obtains a 2 factor approximation. This also tells us that if there exists a polynomial time algorithm for optimally solving integer programs, then the vertex cover is in P. i.e. $\mathsf{P} = \mathsf{NP}$.

2. Theory of Duality

For every linear program there is a dual counterpart.

Definition 2.1. For $\boldsymbol{c}$ and $\boldsymbol{x}$ are n vectors and $\boldsymbol{b}$ and $\boldsymbol{y}$ are m vectors, and $\boldsymbol{A}$ is an $m \times n$ matrix, the dual of a standard maximum problem

$$\begin{aligned} \text{maximize } c^T x \text{ such that} \\ Ax \leq b \\ x \geq 0 \end{aligned}$$

is the standard minimum problem

$$\begin{aligned} \text{minimize } y^T x \text{ such that} \\ y^T A \geq c^T \\ y \geq 0 \end{aligned}$$

We transfer a standard minimum problem to a standard maximum problem by multiplying ***A, b, c*** by -1. We can transform the standard maximum problem similarly back to a standard minimum problem by multiplying by -1. We can see this in this example

Example. Maximize $x_1 + x_2$ for $x_1 \geq 0, x_2 \geq 0$ with constraints:

$$\begin{aligned} x_1 + 2x_2 &\leq 4 \\ 4x_1 + 2x_2 &\leq 12 \\ -x_1 + x_2 &\leq 1 \end{aligned}$$

This corresponds to the standard minimum problem
Minimize $4y_1 + 12y_2 + y_3$ for $y_1 \geq 0, y_2 \geq 0, y_3 \geq 0$ with constraints:

$$\begin{aligned} y_1 + 4y_2 - y_3 &\geq 1 \\ 2y_1 + 2y_2 + y_3 &\geq 1 \end{aligned}$$

Knowing how to transform one form to the another we will now look at two duality theorems.

Theorem 2.2. ***Weak Duality Theorem*** *If the first linear program is in maximization form and the second is in minimization standard form and the two are dual of each other, then,*

- *If the first linear program is is unbounded, then the second is infeasible;*
- *If the second is unbounded, then the first is infeasible;*
- *If the first and second linear programs are both feasible and bounded then (with opt representing optimized value)*

$$opt(LP_1) \leq opt(LP_2)$$

Theorem 2.3. ***Strong Duality Theorem*** *If either first or second is feasible and bounded, then so is the other, and*

$$opt(LP_1) = opt(LP_2),$$

then,

- *If one of first or second is feasible and bounded, then so is its counterpart;*
- *If one of first or second is unbounded, then the counterpart is infeasible;*
- *If one of first or second is infeasible, then the counterpart cannot be feasible and bounded, that is, the other is going to be either infeasible or unbounded. Either case can happen.*

3. Pivot Operation

Before we introduce one of the main algorithms of linear algebra, we will go over a pre-requisite that will allow us to understand the Simplex Method. For any system of equations where we wish to maximize an objective function, there are dependent variables and independent variables. Given a system of n linear functions and m unknowns, we can represent it in matrix form as

$$y^T A = s^T$$

where $y^T = (y_1, ..., y_m)$ and $s^T = (s_1, ..., s_n)$ and A is matrix such that

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

What the pivot operation allows us to do is have a independent variable like y_i and dependent variable s_j be interchanged. The conditions for this is that $a_{ij} \neq 0$. This is because to we derive an equation for y_i by taking the jth equation such that

$$y_i = \frac{1}{a_i j}(-y_1 a_{ij} - \cdots - y_{i-1} a_{(i-1)j} + s_j - y_{i+1} a_{(i+1)j} - \cdots - y_m a_{mj})$$

. Substituting this into the other equations of the system of equations, we arrive to a new system of equations where y_i and s_j have been switched.

$$\begin{array}{ccccccccc} y_1\hat{a}_{11} & + & \cdots & + & s_j\hat{a}_{i1} & + & \cdots & + & y_m\hat{a}_{m1} & = & s_1 \\ \vdots & & & & \vdots & & & & & & \vdots \\ y_1\hat{a}_{1j} & + & \cdots & + & s_j\hat{a}_{ij} & + & \cdots & + & y_m\hat{a}_{mj} & = & y_i \\ \vdots & & & & \vdots & & & & & & \vdots \\ y_1\hat{a}_{1n} & + & \cdots & + & s_j\hat{a}_{in} & + & \cdots & + & y_m\hat{a}_{mn} & = & s_n \end{array}$$

The relation between a_{ij}'s and the $\hat{a}_{ij}$ is shown below for reference, but not discussed in detail as the main focus is to grasp an understanding of what the pivot operation does so we can understand what the Simplex Method does:

$$\hat{a}_{ij} = \frac{1}{a_{ij}}$$
$$\hat{a}_{hj} = -\frac{a_{hj}}{a_{ij}} \text{ for } h \neq i$$
$$\hat{a}_{ik} = \frac{a_{ik}}{a_{ij}} \text{ for } k \neq j$$
$$\hat{a}_{hk} = a_{hk} - \frac{a_{ik}a_{hj}}{a_{ij}} \text{ for } k \neq j \text{ and } h \neq i$$

In short, the pivoting quantity goes to its reciprocal, the entries in the same row are divided by the pivot, the entries in the same column change sign and are divided by the pivot, and the other entries are reduced by a product of the entries that correspond to their row and column divided by the pivot point.

Without the Simplex Method, our only way to maximize/minimize an objective function using pivot operations is to pivot "madly" until all entries in the last column and last row are nonnegative - *an indication that we have maximized/minimized our desired objective function.*

4. The Simplex Method

Created by George Bernard Dantzig, an American mathematical scientist, the simplex method is an incredibly prominent algorithm in linear programming. While its name comes from the concept of a simplex, the actual algorithm involves none. For the simplex method, we will introduce two types of variables *slack variables* and *decision variables*. Slack variables take on the value of our main constraints. Decision variables on the other hand are the original variables in the original canonical linear programming problem. With our slack and decision variables defined, we can then restate the problem from a linear programming problem to optimizing, whether that be maximizing or minimizing, the original objective subject to the slack and decision variables. The main idea behind solving problems with the simplex method is improving successful iterations. Partnered with the theory of duality, one can use the simplex method to solve minimization problems as well. We will first solve a maximization problem using the simplex method to go over basic steps and terminology and then break it down more systematically:

Example. **Solving a Maximization Problem using Simplex Method**

$$3x_1 + 2x_2 + 4x_3$$
$$x_1 + x_2 + 2x_3 \leq 4$$
$$2x_1 + 3x_3 \leq 5$$
$$2x_1 + x_2 + 3x_3 \leq 7$$
$$x_1, x_2, x_3 \geq 0$$

We define our **slack** variables by considering our constraints: since we have three constraints, we will have three slack variables. We define our slack variables using our **decision** variables x_1, x_2, x_3. Let us $define$ our first slack variable $s_1 = 4 - x_1 - x_2 - 2x_3$. This pattern is repeated until we get the three following slack variables of our problem:

$$s_1 = 4 - x_1 - x_2 - 2x_3$$
$$s_2 = 5 - 2x_1 - 3x_3$$
$$s_3 = 7 - 2x_1 - x_2 - 3x_3$$

Then we define a variable z for the expression we wish to maximize. To maximize our objective, we will have to maximize z, such that $x_1, x_2, x_3, s_1, s_2, s_3 \geq 0$. Since the idea of simplex method again is to improve with succession, we will first start by setting our decision variables to 0 and through that evaluate our slack variables to be:

$$s_1 = 4$$
$$s_2 = 5$$
$$s_3 - 7$$

which also gives us a z-value of 0. Let us try something different then, say we keep x_2 and x_3 the same and solve for x_1. We then get the inequality $z = 3x_1 > 0$. Using the condition that our slack variables must be positive we determine: since $s_1 \geq 0$, $x_1 \leq 4$, $s_2 \geq 0$ so $x_1 \leq \frac{5}{2}$, and $s_3 \geq 0$ $x_1 \leq \frac{7}{2}$. This gives us a solution $x_1 \leq frac52$ since it fulfills all of these inequalities. Now the current state of variables are:

$$x_1 = \frac{5}{2}$$
$$x_2 = 0$$
$$x_3 = 0$$
$$s_1 = \frac{3}{2}$$
$$s_2 = 0$$
$$s_3 = \frac{9}{2}$$
$$z = \frac{15}{2}$$

While $z = \frac{15}{2}$ is much better $z = 0$, our solution be further improved by creating a new system. We will now define x_1 in terms of x_2, x_3, s_1:

$$x_1 = 4 - x_2 - 2x_3 - s_1$$

and then substitute x_1 to express our remaining variables in terms of x_2, x_3, s_1:

$$\begin{aligned} s_2 &= 5 - 2(4 - x_2 - 2x_3 - s_1) - 3x_3 \\ &= -3 + 2x_2 + x_3 + 2s_1 \\ s_3 &= 7 - 2(4 - x_2 - 2x_3 - s_1) - x_2 - 3x_3 \\ &= -1 + x_2 + x_3 + 2s_1 \\ z &= 3(4 - x_2 - 2x_3 - s_1) + 2x_2 + 4x_3 \\ &= 12 - x_2 - 2x_3 - 3s_1 \end{aligned}$$

Now looking at our z equation, we observe that the coefficients of x_2, x_3, s_1 which would decrease our z value. So we have found our max z value which is 12, and our solution is:

$$\begin{gathered} x_1 = 4, x_2 = 0, \\ x_3 = 0, s_1 = 0, \\ s_2 = -3, s_3 = -1, \\ z = 12 \end{gathered}$$

4.1. **The Simplex Method Systematically.** The Simplex Method helps us to choose pivots to approach the solution systematically. If we pivot for awhile and get the the tableau where $\mathbf{b} \geq 0$

	r	
t	***A***	***b***
	-c	***v***

We reach a **feasible point** for a maximum problem when we let $\mathbf{r} = \mathbf{0}$ and $\mathbf{t} = \mathbf{b}$. For a minimum point, if **-c** ≥ 0, we have a feasible point when $\mathbf{r} =$ **-c** and $\mathbf{t} = \mathbf{0}$. Our first case is when we have already found a feasible point for the maximum problem $\mathbf{b} \geq 0$.

Case 1: $\mathbf{b} \geq 0$ Choose a column j_0 that has a last entry negative column. For the row, choose the one that has the smallest $b_i/a_{i,j_0}$ ratio and if there are any ratios, choose any such i_0. This will be our new pivot point. When you can no longer pivot when $\mathbf{b} \geq 0$, there are two possibilities. One, for all the columns the last column entry $-c_j \geq 0$, which means you have found the solution - v being the value of your program. Two, your maximum problem is unbounded feasible as you find that some last entry columns are less than 0 and that for all $a_{i,j_0} \leq 0$ for all i. This rule works for selecting our pivot point because the **b** column continues to be nonnegative, so we still have a feasible point and the new tableau is generally greater than the old. Therefore, if we continue to use simplex method you will either find the maximum or that the problem is unbounded feasible.

Case 2: Some b$_i$ are negative. In this case we find the first negative b_i - b_k and find any negative entry in the row of k. We will then compare $b_k/a_{k,j_0}$ and $b_i/a_{i,j_0}$ which $b_i \geq 0$ and $a_{i,j_0} \geq 0$ and choose the i_0 that gives the smallest ratio, and this would be the pivot point. The only way the rules of case 2 do not apply is if we cannot find any negative entries for the negative b values, and if this is the case, then the maximum problem is infeasible.

5. The Ellipsoid Method

5.1. **The Klee Minty Example.** Even though the simplex algorithm does well in most of the cases, it can do as bad as exponential in some. We consider the following instance of the problem with n variables and n constraints.

$$\max\ 2^{n-1}x_{n-1} + 2^{n-2}x_{n-2} + \cdots + 2x_1 + x_0$$
$$x_{n-1} \leq 5$$
$$4x_{n-1} + x_{n-2} \leq 25$$
$$8x_{n-1} + 4x_{n-2} + x_{n-3} \leq 125$$
$$\vdots$$
$$2^n x_{n-1} + 2^{n-1}x_{n-2} + \cdots + x_0 \leq 5^n$$
$$x_i \geq 0\ \forall i$$

The elementary simplex method goes through all the 2^n extreme points before reaching the optimal solution: $(0, 0, \cdots, 5^n)$.

5.2. **Reducing to a Promise LP.** This algorithm solves an equivalent (as shown below) problem known as a Promise Linear program in polynomial time. The idea is to consider an *ellipsoid* which contains the set of feasible solutions and try to restrict the volume by cutting it in half in a subsequent iteration.

Definition 5.1. An ellipsoid is the generalization of an ellipse to n dimensions. The set of points $(x_1, x_2, \cdots, x_n)$ forming an ellipsoid centered at the origin in n dimensions satisfy the following equation:

$$\sum_{i=1}^{n} \frac{x_i^2}{a_i^2} \leq 1$$

where the a_i's are the lengths of the semi-axes. An equivalent definition using matrices is as follows. Let $A = \begin{bmatrix} \frac{1}{a_1^2} & 0 & \cdots & 0 \\ 0 & \frac{1}{a_2^2} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \frac{1}{a_n^2} \end{bmatrix}$. Then an ellipsoid can be given by the set of points

$$\{(x_1, x_2, \cdots, x_n) | xAx^T \leq 1\}.$$

Definition 5.2 (Bounded Linear Program)**.** Is there a a vector $\mathbf{x} = [x_1, x_2, \cdots, x_n]$ such that

$$Ax \leq \mathbf{b},$$
$$x_i \geq 0,$$
$$x_i \leq c\ \forall i,$$

where A is a matrix, b a vector and c a natural number.

We briefly show the ideas behind the reduction from a Standard LP to a Promise LP by showing that a standard one reduces to a bounded instance - which in turn, reduces to a promise instance.

Theorem 5.3. *A standard LP instance can be reduced to a promise LP instance.*

Proof. Consider a standard LP. We will remove the maximizing condition and add a bound to convert it into a bounded LP. As in the notation defined at the beginning, denote by M the maximum absolute value from all the integers in $A, \mathbf{b}, \mathbf{c}$. Instead of the maximising condition, we add

$$(nk)^{n^2} \leq c^T x \leq (nk)^{n^3}.$$

Further, the feasibility region can be bounded by

$$(nM)^{n^3}[1, 1, \cdots, 1].$$

We run the polynomial time algorithm on this bounded instance. We now have two cases to consider.
Case 1. There is such an $\mathbf{x}$. It can be shown that this implies that the original standard LP is *unbounded* and thus a maximum cannot be attained.
Case 2. There is no such $\mathbf{x}$. Consider a new bounded LP instance created by removing the maximizing condition from the standard LP and adding the constraint

$$\mathbf{x} \leq (nM)^{n^2}[1, 1, \cdots, 1].$$

It can be shown that there is an optimal solution to the original standard LP if and only if there is a solution to this newly created bounded LP. ■

Definition 5.4 (Promise Linear Program)**.** With notation as in **Definition 4.4**, we ask the following question: is there a vector $\mathbf{x} = [x_1, \cdots, x_n]$ such that

$$Ax \leq \mathbf{b},$$
$$x_i \geq 0, \text{ and}$$
$$x_i \leq (nM)^{n^2} \ \forall i?$$

Theorem 5.5. *A promise LP instance can be reduced to a bounded LP instance.*

Proof. Consider an instance of the bounded LP problem with the constraint $A\mathbf{x} \geq \mathbf{b}$. It is possible to slightly tweak this LP so that it is of the form

$$A\mathbf{x} \geq \mathbf{b} - \varepsilon\mathbf{e}, \text{ where } \mathbf{e} = [1, 1, \cdots, 1].$$

There are two parts of the proof. We mention brief ideas for the same.
Part 1: showing that the new LP is an instance of a Promise LP.
We let $\mathbf{x}$ be a feasible solution to $A\mathbf{x} \geq \mathbf{b}$. Setting $\mathbf{y}$ such that

$$|y_i - x_i| \leq \frac{\varepsilon}{nM} \ \forall i.$$

Upon some algebra (and choosing a satisfactory ε) this constraint ensures that $A\mathbf{y} \geq \mathbf{b} - \varepsilon\mathbf{e}$. Further, because of the constraint, the set of all such y forms a cube of side $\varepsilon/(nM)$. It can be shown that this cube contains a sphere of radius $\frac{1}{(nM)^{n^3}}$. This shows that the new Linear Program mentioned above is a Promise LP.
Part 2: showing that the new LP has a solution if and only if the original bounded one does. It is easy to see that if $\mathbf{x}^*$ is a solution to the bounded LP, then $\mathbf{x}^* + \varepsilon\mathbf{e}$ is a solution to the promise LP. The other way round is slightly tricky - and can be shown by proving the contra positive. i.e. if the bounded LP has no solutions, so does the promise LP. The bounded LP can be rewritten by adding an objective function: maximizing $\mathbf{0}^T\mathbf{x}$ with $A\mathbf{x} \geq \mathbf{b}$. Doing some algebra can after assuming that the promise LP is feasible leads to a contradiction. We do not mention the entire proof here. ■

Let us now restate the problem we will be solving before getting to the algorithm.

Question 5.6. *Given a matrix A and vector* $\mathbf{b}$ *with integer entries, is there a vector* $\boldsymbol{x}$ *satisfying*

$$A\boldsymbol{x} \leq \mathbf{b},\ x_i \geq 0,\ x_i \leq (nM)^{n^2}$$

where M is the maximum absolute value from all the integer entries in the given vectors/matrices. Further, should the answer be yes, the feasible region contains a sphere of radius $\frac{1}{(nM)^{n^3}}$

5.3. **Algorithm Idea.** Our first observation is that each variable lies between 0 and $(nM)^{n^2}$ - i.e. we can say that the feasible region lies entirely inside a sphere of radius $(nM)^{n^3}$ centered on around the origin. We set this as our Ellipsoid for this iteration. Now, if the center of this Ellipsoid (the origin in this case) is feasible, then clearly we are done - we return it and stop the algorithm. If it is not feasible, then there exists a constraint which was not satisfied by it. We construct a half plane passing through the center so that the violated constraint is now satisfied. We now have a restricted region containing the feasible points.

Theorem 5.7. *Let E be an ellipsoid and P be a plane passing through the center of E which cute the ellipsoid into two parts. There exists another ellipsoid which entirely contains the points lying in the intersection of half plane (formed by P) and E.*

Let E' be the new ellipsoid obtained from this theorem. It can also be shown (we do not go into much details here) that if the given LP is feasible, then the volume of the new ellipsoid must lie above a threshold. More precisely, if the algorithm runs for more than $2(n+1)n^4 \ln(nM)$ iterations, the corresponding ellipsoid won't contain a sphere of radius $\frac{1}{(nM)^{n^3}}$ and hence the LP won't be feasible.

The running time of this algorithm can be shown to be $O(n^6 L)$, where n is the number of variables and L is the number of bits to the input.

6. Comparing Efficiencies

Since the problem was introduced, there has been a lot of progress and a lot of variants of the methods we saw above have come out. There are variants of the simplex method that do well and are still used in practise. The Ellipsoid algorithm, being the first polynomial time algorithm to solve the problem takes a huge amount of time ($O(n^6)$, as seen) on all possible inputs and thus is not feasible in practise. Another huge breakthrough came out in the 1980s - when mathematician Narendra Karmakar came up with the interior point method. It combines the workings of the Simplex method and Ellipsoid algorithm to come up with a $O(n^3)$ time way to solve the problem. It was observed to do as well as the simplex method in practise and is also used in a lot of implimentations today.

7. References

- The Ellipsoid Method: https://www.iitgoa.ac.in/ sreejithav/misc/ellipsoid.pdf
- Theory of Duality in Linear Programming: https://theory.stanford.edu/ trevisan/cs261/lecture06.p
- Linear Programming: https://archive.org/details/linearprogrammin00chv/page/n3/mode/2up
- Simplex Method: https://www.usna.edu/Users/weaprcon/avramov/$_files/documents/compressed-sensing-tutorial/LP.pdf$