Section Notes 2

# The Linear Programming Model

Applied Math / Engineering Sciences 121

Week of September 19, 2016

**TF VERSION - Do Not Distribute**

## Goals for the week

1. be able to put any linear program (LP) into canonical form.
2. understand the relationship between basic solutions and extreme points.
3. know what non-linear programs can be transformed into linear programs and how to do so.
4. be comfortable with basic AMPL syntax relevant to defining LPs.

## Contents

## 1 The Objective Function and Solutions for LPs

Consider the following linear program:

$$\begin{array}{rc} \text{maximize} & c_1x_1 + c_2x_2 \\ \text{subject to} & x_2 \geq 1 \\ & x_1 + x_2 \geq 2 \end{array}$$

for nonzero constants $c_1$ and $c_2$.

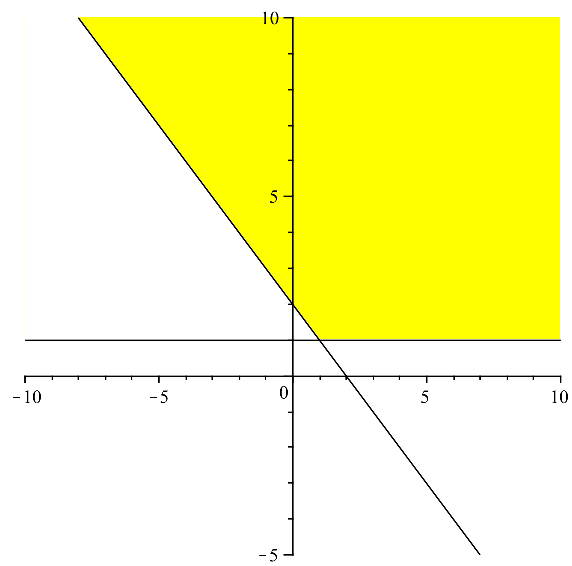


Figure 1: The feasible region

**Exercise 1**

Can you find an example of values for $c_1$ and $c_2$ such that the problem

1. has a single optimal solution?
2. has exactly two optimal solutions?
3. has an infinite number of optimal solutions.
4. becomes infeasible?

**End Exercise 1**

**Solution 1**

To get a feel for the problem, we can first graph the feasible region (done in Maple here): The resulting graph is shown in Figure 1.

1. $(c_1, c_2) = (-1, -2)$. The slope of the objective function (-0.5) is between the slopes of the constraints (-1 and 0). Graphically, the vector (-1,-2) is orthogonal to the contour lines.
2. No, that won't be possible. Because we have linear constraints and a linear objective, if we find two optimal solutions we automatically have an infinite number of optimal solutions.
3. $(c_1, c_2) = (0, -1)$. Since the objective has the same slope as the horizonal bounds for the feasible region, the optimal objective value is reached by all points on the southern frontier of the feasible region, i.e. we have an infinite number of optimal solutions. Through similar reasoning, $(c_1, c_2) = (-1, -1)$ would induce an infinite number of optimal solutions as well.
4. No. In fact, posing the question this way does not make a lot of sense. Feasibility or infeasibility does not depend on the objective function but is determined by the constraints.

**End Solution 1**

By completing the above exercises, you should now:

- understand the relationship between the objective function, constraints, feasibility, and the number of optimal solutions.
- be comfortable identifying coefficients that will change the number of optimal solutions.

## 2 The Standard Form and the Canonical Form

In standard form, a linear program can be defined as follows:

$$\max \mathbf{c}^\mathbf{T}\mathbf{x} \tag{1}$$
$$A\mathbf{x} \leq \mathbf{b} \tag{2}$$
$$\mathbf{x} \geq 0 \tag{3}$$

In canonical form, a linear program looks like the following one:

$$\begin{aligned} \text{maximize} \quad & 0x_1 + 0x_2 - 3x_3 - x_4 + 20 \\ \text{subject to} \quad & x_1 - 3x_3 + 3x_4 = 6 \\ & x_2 - 8x_3 + 4x_4 = 4 \\ & x_i \geq 0, (i = 1, 2, 3, 4) \end{aligned}$$

Note that this terminology are not universal and different textbooks may refer to them differently.

**Exercise 2**

What are the defining properties of an LP in canonical form?

**End Exercise 2**

**Solution 2**

- It is a maximization problem.
- All decision variables are non-negative.
- All constraints are equalities.
- The RHS coefficients are all non-negative.
- One decision variable is "isolated" in each constraint with a +1 coefficient. These variables do not appear in any other constraint and have a zero coefficient in the objective function.

**End Solution 2**

**Exercise 3**

Is the canonical form restrictive in any way?

**End Exercise 3**

**Solution 3**

No, it's not. We can transform any LP into canonical form, provided we are willing to use artificial variables (see lecture notes for lecture 2). Give short example for a) minimization, b) negative RHS, c) non-equality constraints, d) negative decision variables, e) non-isolated variables.

**End Solution 3**

**Exercise 4**

What are the advantages of having an LP in canonical form?

**End Exercise 4**

**Solution 4**

We can easily find a feasible solution for the LP simply by setting all isolated variables equal to the RHS of the corresponding constraint.

**End Solution 4**

**Exercise 5**

Convert the following LP to canonical form:

$$\begin{aligned} \text{minimize} \quad & x_1 + 3x_2 \\ \text{subject to} \quad & 2x_1 + x_2 \geq -5 \\ & x_1 - x_2 \leq 6 \\ & x_i \geq 0, (i = 1, 2) \end{aligned}$$

**End Exercise 5**

**Solution 5**

$$
\begin{aligned}
\text{maximize} \quad & -x_1 - 3x_2 \\
\text{subject to} \quad & -2x_1 - x_2 + x_3 = 5 \\
& x_1 - x_2 + x_4 = 6 \\
& x_i \geq 0, (i = 1, 2, 3, 4)
\end{aligned}
$$

**End Solution 5**

By completing the above exercises, you should now:

- know the definition and properties of canonical form.
- be able to put any LP into canonical form.

# 3 Basis, Variables, Extreme Points and Solutions

## 3.1 Review

Linear algebra review:

1. A **basis** of $R^m$ is a set of linearly independent $m$-dimensional vectors $v_1, ..., v_m$ with the property that every vector of $R^m$ can be written as a linear combination of the vectors $v_1, ..., v_m$. Note that the vectors $v_1, ..., v_m$ form a square matrix that is invertible. These vectors $v_1, ..., v_m$ **span** the vector space $R^m$.
2. A **basis** $B$ for an arbitrary $m$-by-$n$ matrix A can also be seen as a list of $m$ numbers chosen from {1, 2,. . . , n} such that the square matrix $A_B$ with $m$ columns from $A$ indexed by this list is a basis for $R_m$, i.e. the column vectors **span** the space $R_m$. Again, $A_B$ will be invertible.

Basic feasible solutions and extreme points

1. **Basic variables** for a given basis $B$ are the ones corresponding to the column vectors of $B$.
2. **Non-basic variables** for a given basis $B$ are all variables except the **basic variables**.
3. The **basic solution x** of the system $Ax = b$ for a basis $B$ is the unique solution of this system where all non-basic variables are equal to zero ($x_j = 0$ for all indices $j \notin B$).
4. A **feasible solution** is a solution for an LP which satisfies all the constraints.
5. A **basic feasible solution** is a solution for an LP that is both basic and feasible. Note that basic solutions for LPs in canonical form are solutions for $Ax = b$ but they might be infeasible if the non-negativity requirements for the decision variables are not satisfied.
6. Let $S$ be the set of points in the feasible region of an LP. A point $y$ in $S$ is called an **extreme point** of $S$ if $y$ cannot be written as $y = \lambda w + (1 - \lambda)x$ for two distinct points $w$ and $x$ in $S$ and $0 < \lambda < 1$. That is, $y$ does not lie on the line segment joining any two points of $S$.
7. Every **basic feasible solution** of an LP corresponds to an **extreme point** of the feasible region of the LP.

### 3.2 Finding Basic, Feasible Solutions and Extreme Points

Example: Consider the following two constraints:

$$x_1 + x_2 = 2$$
$$2x_1 + 2x_2 = 3$$

**Exercise 6**

Can we find a vector $(x_1, x_2)$ that satisfies both constraints, i.e. a basic solution?

**End Exercise 6**

**Solution 6**

No, we cannot. Note that the two column vectors of matrix $A$ are linearly dependent and thus do not span $R^2$ and no linear combinations of the column vector gets us the RHS vector.

**End Solution 6**

Now, consider the set of vectors $x$ satisfying the following constraints:

$$\begin{aligned} x_1 + 2x_2 + 2x_3 &= 4 \\ 2x_1 + 4x_2 + 3x_3 &= 7 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

**Exercise 7**

1. List the basic solutions corresponding to these equations. For each basic solution you write down, specify the columns of the coefficient matrix corresponding to the solution. Which ones are feasible?
2. Find an extreme point of the feasible region.
3. Find a feasible solution that is not an extreme point.

**End Exercise 7**

**Solution 7**

1. We start by noting that the columns of the coefficient matrix span. We consider the set of pairs of indices for a basis, and check whether the corresponding columns are linearly independent. The first and the second column are linearly dependent, and thus not a basis. The first and the third column are linearly independent and thus form a basis, same for the second and third column. We can compute the corresponding basic solutions:
   - $\{1, 3\} \Rightarrow (2, 0, 1)$. This is a feasible solution.
   - $\{2, 3\} \Rightarrow (0, 1, 1)$. This is feasible.

2. Since the basic feasible solutions correspond to the extreme points of the feasible region, both basic feasible solutions found are such points.
3. By taking a linear combination of two extreme points we get a feasible solution that is not an extreme point. Thus: $\frac{1}{2} \cdot (2, 0, 1) + \frac{1}{2} \cdot (0, 1, 1) = (1, \frac{1}{2}, 1)$.

**End Solution 7**

By completing the above exercises, you should now:

- understand the relationship between basic solutions and extreme points.
- know how to obtain all basic solutions for a given LP by putting it into canonical form.

### 3.3 Learning to pivot

The Simplex algorithm iterates from tableau to tableau by performing pivot operations. In the following exercise, we will get some practice with pivoting.

In geting started, we will not work on the full tableau of the Simplex algorithm. Rather, we will pivot on a matrix. We can think about this matrix as representing the $m$-by-$n$ $\bar{A}$ matrix in the tableau of the simplex algorithm. It does not include the right-hand side vector $\bar{b}$ or the objective equation that is part of the simplex method's tableau.

First, two definitions:

- Let $e_k$ be the $m$-dimensional unit column vector $(0, \ldots, 1, \ldots, 0)$ with a 1 in the $k$-th place and 0's elsewhere. The $m$-by-$n$ **matrix $A$ is in canonical form** if all the vectors $e_k$ occur as columns of $A$.
- **Pivoting** is the process of applying algebraic manipulations (multiplying rows with scalars, adding rows to other rows) to a matrix such that we get one in canonical form. One **pivot operation** refers to the series of algebraic manipulations to get one new unit vector $e_k$ in the matrix that wasn't there before.

Now, consider the following matrix:

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \end{pmatrix}$$

**Exercise 8**

1. Using 2 pivot operations, arrive at a matrix in canonical form (i.e., with two unit vectors).
2. Using this matrix, find one other corresponding matrix in canonical form by using a single pivot operation.

**End Exercise 8**

**Solution 8**

1. (1). Subtract two times row 1 from row 2. (2). Multiply row 2 by -1. Subtract two times row 2 from row 1. Thus, we get:

$$\begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \end{pmatrix}$$

2. We pivot on the third entry in row 1: (1). Add two times row 1 to row 2. (2). Multiply row 1 with -1. Thus we get:

$$\begin{pmatrix} -1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}$$

**End Solution 8**

By completing the above exercises, you should now:
- know how to use pivot operations to arrive at matrix in canonical form.
- know how to use pivots to bring in a new unit vector.

**Exercise 9**

Transform the following non-linear program into an L.P.

$$\begin{aligned} \text{minimize} \quad & \mid x_1 \mid + \mid x_2 \mid + \mid x_3 \mid + \mid x_4 \mid + \mid x_5 \mid + \mid x_6 \mid + \mid x_7 \mid \\ \text{subject to} \quad & x_1 + x_4 + x_7 \geq 5 \\ & x_1 + x_2 + x_3 + x_6 \geq 8 \end{aligned}$$

Hint: Consider the graph $\mid x_1 \mid$ and determine how the maximum of two or more linear functions could generate the same graph.

**End Exercise 9**

**Solution 9**

Note: $\mid x_i \mid = \max \{x_i, -x_i\}$ for each $i$. We create a new variable $z_i$ for each term in the sum and append the constraints such that $z_i \geq x_i, z_i \geq -x_i \; \forall i$

$$\begin{aligned} \text{minimize} \quad & z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 \\ \text{subject to} \quad & x_1 + x_4 + x_7 \geq 5 \\ & x_1 + x_2 + x_3 + x_6 \geq 8 \\ & z_i \geq x_i \; \forall i \in \{1, ..., 7\} \\ & z_i \geq -x_i \; \forall i \in \{1, ..., 7\} \end{aligned}$$

**End Solution 9**

# 4 Modeling with AMPL - The Street Illumination Problem

## 4.1 Optimal Street Illumination Problem

Suppose we have a street that is five units long. At each unit, there is a lamp post with a dimmable lamp (i.e. we can control the power used and thus light generated at each post). At each of the three middle units of the street there is a store, and these stores have minimum lighting requirements of 5, 3 and 7 respectively. Each street light casts a beam over its own area of the street, as well as the areas to each side in a proportion of 10%, 80%, 10% of the total light produced by the lamp burning at a specific intensity. Your goal: light the street by at least as much as is required by the store owners, while using as little power as possible.

Given this setup, let's use the new techniques to make an elegant model for this problem:

**Exercise 10**

1. What are the sets implicit in the problem?
2. What are the decision variables?
3. What is the objective function?
4. What are the constraints? Hint: Think about how can we use offset indexing to elegantly describe the constraints
5. What if instead of wanting to minimize the total power used, the store owners wanted to minimize the difference between the required power and the power received? Give all the relevant changes to the LP if the goal is to minimize the absolute difference between the power required and the power received.

**End Exercise 10**

**Solution 10**

1. The STREET itself, the LAMPS (the same given above setup, but it could have been setup differently), the STORES area with a minimum required light, the CAST pattern of each light
2. One variable for the power from each lamp post, $p_i$
3. $\min \sum_{i \in LAMPS} p_i$
4. We can think about the constraints from the point of view of the three areas in front of the stores – that is where we have a constraint. We need to walk the STORES area, and for each location make a constraint that ensures that the total light emanating from all lamps that CAST upon that location is at least the required value.
   The resulting constraints are:
   $.8p_i + .1p_{i-1} + .1p_{i+1} \geq r_i, \ \forall i \in \{STORES\}$ (total power on area i is greater than or equal power required for area i)
   $p_i \geq 0 \ \forall i \in \{LAMPS\}$ (non-negativity constraint)

5. With this new objective, the required light constraints have been softened and it is no longer necessary to meet the desired amount but to keep the amount of power on each spot as close to the amount (either above or below) as possible. Because it is asking about absolute difference it is clear we need to use absolute values. If we were going to solve a non-linear LP we would write the following program:

$$\begin{array}{ll} \text{minimize} & \sum \mid .8x_i + .1x_{i-1} + .1x_{i+1} - r_i \mid \ \forall i \in \{STORES\} \\ \text{subject to} & x_i \geq 0 \ \forall i \in \{LAMPS\} \end{array}$$

To make this program linear, we need introduce a new variable $z_i$, $\forall i \in \{STORES\}$ that represents the absolute difference between the total power on each area and the required power. We can do this by requiring $z_i$ to be greater than both the difference between the total power put on an area and the required and vice versa and minimize the sum of $z_i$ (similar to above). The final LP would be:

$$\begin{array}{ll} \text{minimize} & \sum z_i, \ \forall i \in \{STORES\} \\ \text{subject to} & z_i \geq .8x_i + .1x_{i-1} + .1x_{i+1} - r_i, \ \forall i \in \{STORES\} \\ & z_i \geq r_i - (.8x_i + .1x_{i-1} + .1x_{i+1}), \ \forall i \in \{STORES\} \\ & x_i \geq 0, \ \forall i \in \{1, ..., 7\} \end{array}$$

**End Solution 10**

## 4.2 A Review of AMPL syntax

As part of your current problem set, you will need to be able to create matrices of both variables and parameters in your AMPL code.

**Exercise 11**

1. What is the syntax for declaring a parameter?
2. What is the syntax for declaring a vector of parameters?
3. What is the syntax for declaring a matrix of parameters?
4. What is the syntax for declaring a variable?
5. What is the syntax for declaring a vector of variables?
6. What is the syntax for declaring a matrix of variables?
7. What is the syntax for defining a parameter?
8. What is the syntax for defining a vector of parameters?
9. What is the syntax for defining a matrix of parameters?

**End Exercise 11**

**Solution 11**

1. `param foo >= 0;`
2. `param foo{SET} >= 0;`
3. `param foo{SET_A, SET_B} >= 0, <= 1;`
4. `var Bar >= 0;`
5. `var Bar{SET} >= 0;`
6. `var Bar{SET_A, SET_B} >= 0;`
7. `param foo 0.376;`
8. ```
   param:  foo :=
   E1 5
   E2 6
   E3 7 ;
   ```
9. ```
   param foo:  A1 A2 :=
   B1 1 2
   B2 3 4 ;
   ```

**End Solution 11**

By completing the above exercises, you should now:
- be comfortable with basic AMPL syntax relevant to defining LPs.

## 4.3 The Production Model in AMPL

In this example we have a set of products that use up resources to manufacture and store. Specifically the resources are machine time and storage space. Each product fetches a certain profit per unit sold. The producer wants to maximize profit. Let us formulate the model:

| | | |
|---|---|---|
| Sets | $P$ | set of product types |
| | $R$ | set of resources |
| Parameters | $a_i, i \in R$ | amount of resource $i$ available |
| | $p_j, j \in P$ | per unit profit for product $j$ |
| | $u_{ij}, i \in R, j \in P$ | amount of resource $i$ required per unit production of $j$ |
| Variables | $x_j, j \in P$ | quantity of product $j$ to produce |

$$\max \sum_{j \in P} p_j x_j \qquad \text{maximize total profit}$$

$$\text{s.t.} \quad \textstyle\sum_{i \in R} u_{ij} x_j \leq a_j \quad \forall i \in R \qquad \text{resource constraints}$$

We are given the following data. There are 2 products $A$ and $B$. Product $A$ earns a profit of 5 per unit sold, and uses 1 unit of machine time and 5 units of storage space. Product $B$ earns a profit of 8 per unit sold, and uses 1 unit of machine time and 9 units of storage space. The total machine time available is 6 and the total storage space is 45. Using this data to fill in the sets and parameters we have defined we get: $P = \{A, B\}$, $R = \{\text{machine, space}\}$, $a = \{6, 45\}$, $p = \{5, 8\}$, $u_{\text{machine}} = \{1, 1\}$, $u_{\text{space}} = \{5, 9\}$. Here we have assumed that the parameter values over sets are ordered according to the order of the elements appearing in the set (e.g. $A$ before $B$). We can then write the linear program as follows:

$$\begin{aligned} \max 5x_1 + 8x_2 \\ s.t. : x_1 + x_2 \leq 6 \\ 5x_1 + 9x_2 \leq 45 \\ x_i \geq 0, (i = 1, 2) \end{aligned}$$

Here is what the example model and data files would look like in AMPL:

**File: example.mod**

```
set PRODUCT;
set RESOURCE;

param usage {i in RESOURCE, j in PRODUCT};
param profit {j in PRODUCT};
param avail {i in RESOURCE};

var X {j in PRODUCT}>=0;
maximize Total_Profit: sum {j in PRODUCT} profit[j]*X[j];
subject to Resource_Constraints {i in RESOURCE}: sum{j in PRODUCT}
                                                 usage[i, j]*X[j]<=avail[i];
```

**File: example.dat**

```
set PRODUCT := A        B;
set RESOURCE := machine space;

param usage:   A        B:=
        machine 1       1
        space   5       9;
param profit:= A        5       B       8;
param: avail:= machine 6        space 45;
```

### 4.4 Street Illumination, in AMPL

**File: light.mod**

```
param street_end >= 1, integer;        # Street length
param cast_size >= 1, integer;         # Size of the light cast

set STREET := 1..street_end;           # Positions on the street
set CAST := -cast_size..cast_size;     # Offsets for the lights

param light_locations{STREET} >=0, <=1, integer;
                                       # 1 where the lights are, 0 otherwise
param light_required{STREET} >=0;      # where we need light
param intensity{CAST};                 # Pattern of light intensity

set LIGHTS := {i in STREET: light_locations[i] > 0};
                                       # Where the lights are
set ILLUMINATED := {i in STREET: light_required[i] > 0};
                                       # Where we need light

var Power{LIGHTS} >= 0;                # How much power to apply to each light

minimize Total_Power: sum{l in LIGHTS} Power[l];
                                       # Minimize total power usage

subject to Required_Light {i in ILLUMINATED}:
                          sum{l in LIGHTS, c in CAST: i=l+c}
                             intensity[c] * Power[l]
                          >= light_required[i];
                                       # Ensure sidewalks get required light
```

**File: light.dat**

```
param street_end := 5;                 # Street length
param cast_size := 1;                  # Size of illumination will be -cs...+cs

param: light_locations light_required :=
1      1               0               # Where the lights are [0,1]
2      1               5               # And where we need light
3      1               3
4      1               7
5      1               0 ;

param: intensity :=                    # Pattern of light intensity
-1     .1
0      .8
1      .1 ;
```

## 4.5 A Review of Set Notation

Set notation is a way of describing the membership in and relationships between groups of objects. A set is a finite or infinite collection of objects in which order has no significance. Members of a set are referred to as elements and the notation $a \in A$ is used to denote that $a$ is an element of a set $A$. The elements of a set can be anything: numbers, people, letters of the alphabet, etc. Sets are usually denoted with capital letters. The statement that sets $A$ and $B$ are equal means that they have precisely the same members.

### 4.5.1 Defining sets

A set $A$ can be defined as follows:

$$A = \{2, 4, 6, 8\}$$

While this works for simple sets, it may be more practical to use functions, equalities or other appropriate devices as shown below:

$$\begin{aligned} B &= \{x : 13 \leq x \leq 29; x \text{ is integer}\} \\ C &= \{x : 0 < x < 10; x \bmod 2 = 0; x \text{ is integer}\} \end{aligned}$$

An example of an infinite set is the set of all integers (positive and negative). This set is defined as follows:

$$Z = \{..., -2, -1, 0, 1, 2, ...\}$$

We can also define sets based on conditionals of other sets. For example we can define a new set consisting of the set of all elements of $\mathbb{Z}$ such that their value is greater than some integer $k$:

$$F = \{x : x \in Z; x > k\}$$

Set $F$ is a *subset* of set $Z$. Subsets are sets which are formed from the members of an existing set. In mathematical notation this is denoted as: $F \subset Z$.

### 4.5.2 Set Intersection

An intersection operation between two sets produces a result which contains elements which are common to both sets. If set $M = \{x\colon x \in Z;\ x > 0;\ x \text{ odd}\}$ and set $L = \{x\colon x \in Z;\ x \bmod 5 = 0\}$ then $M \cap L = \{x\colon x \in Z;\ x > 0;\ x \text{ odd};\ x \bmod 5 = 0\}$.

### 4.5.3 Set Union

A union operation between two sets produces a result which contains the members of both sets. If set $K$ = {red, green, blue} and set $H$ = {blue, white, yellow, black} then $K \cup H$ = {red, green, blue, white, yellow, black}.

### 4.5.4 Working with Sets

**Exercise 12**

1. Given the sets $A$ and $C$ defined above, are the sets $A$ and $C$ equal?
2. Given the sets $Z$ and $F$ defined above, what is $Z \cap F$?
3. How would you define the set of even numbers less than 100?
4. How would you define the set of all elements on the diagonal of a matrix?

**End Exercise 12**

**Solution 12**

1. Yes!
2. Since $F$ is a subset of $Z$ the intersection of the two sets is $F$.
3. set $E = \{x\text{: } x \in Z; x \bmod 2 = 0; x < 100\}$
4. set $G = \{(i, j)\text{: } i = j\}$

**End Solution 12**

By completing the above exercises, you should now:

- know how to define sets.
- understand set operations such as set intersection and union.