

Introduction to Optimization

Apurva Nakade

2022-07-11

Contents

1	Introduction	7
I	Simplex Method	11
2	Standard Linear Program	13
3	The Simplex Method	17
3.1	The Simplex Step	19
3.1.1	Entering variable	19
3.1.2	Leaving Variable	20
3.2	Tableau Notation	21
4	Initialization	23
4.1	Auxiliary Linear Program	23
4.2	Combined tableau	25
5	Cycling	29
5.1	Degeneracy	30
5.2	Bland's Rule	32
6	Standardization	33
6.1	Equivalence of Linear Programs	34
II	Duality Theory	37
7	Dual Linear Program	39
7.1	General Linear Program	40
8	Weak and Strong Duality	43
8.1	Weak Duality	43
8.2	Strong Duality	44
8.3	Complimentary slackness	45

9	Sensitivity Analysis	49
9.1	Dictionaries Revisited	49
9.2	Range of Optimality - Constraints	51
9.3	Shadow Prices	53
9.4	Sensitivity analysis - Objective	55
III	Non-Linear Programming	59
10	Convex Programming	61
11	Separation Theorems	65
11.1	Farkas' Lemma	65
11.2	Separating Hyperplane Theorem	66
11.3	Equivalence with Strong Duality	67
12	Interior Point Methods	71
12.1	Gradient Descent	71
12.2	Interior Point Method	72
13	KKT conditions	75
IV	Applications	79
14	L1-Regression	81
15	Network Flow	83
15.1	Min-cut	83
16	Integer Programming	85
16.1	Branch and Bound	86

Preface

These are class notes for a quarter long course on Introduction to Optimization (Math 368) at Northwestern University taught during the Winter and Spring of 2022. For the most part, these notes talk about linear programming and duality theory, and only towards the very end we discuss non-linear programming. The class is aimed at upper-level undergrads who've completed a basic sequence in linear algebra and calculus. The prerequisites for the linear programming part are Gaussian elimination and basic theorem proving. For the non-linear programming part, you'll need to know basic properties of gradients and directional derivatives.

These notes are licensed under a Creative Commons Attribution-ShareAlike 4.0 International License. Please feel free to fork and modify the Github repo or use these notes as-is. Any suggestions and improvements to the notes are welcome and greatly appreciated. You can send me an email or open an issue on Github.



Chapter 1

Introduction

An optimization problem consists of maximizing or minimizing a real function over a set of input values from within an allowed set.

$$\begin{array}{ll} \text{optimize: } & f(x_1, \dots, x_n) \\ \text{subject to: } & (x_1, \dots, x_n) \in D. \end{array}$$

These kinds of questions arise in all quantitative disciplines from computer science and engineering to operations research and economics. As stated, this question is too broad. To make any meaningful analysis, we need to make several simplifying assumptions about f and D .

To begin with, we'll assume that the function f is linear and the constraint set D is described using linear inequalities. The study of this problem is called Linear Programming. Despite the simplicity of the linear programming setup, or perhaps because of it, LP is one of the most commonly used models for optimization problems. Let's start with an example before getting into the precise definitions.

Example 1.1. A bond portfolio manager has \$100,000 to allocate to two different bonds; one corporate and one government bond. The corporate bond has a yield of 4%, a maturity of 3 years and an A rating from a rating agency that is translated into a numerical rating of 2 for computational purposes. In contrast, the government bond has a yield of 3%, a maturity of 6 years and a rating of Aaa with the corresponding numerical rating of 1 (lower numerical ratings correspond to higher quality bonds). The portfolio manager would like to allocate her funds so that the average rating for the portfolio is no worse than Aa (numerical equivalent 1.5) and average maturity of the portfolio is at most 3.6 years. Any amount not invested in the two bonds will be kept in a cash account that is assumed to earn no interest for simplicity and does not contribute to the average rating or maturity computations. How should the manager allocate her funds

between these two bonds to achieve her objective of maximizing the annual yield from this investment? [Cornuéjols et al., 2018]

	Corporate	Government	Constraints
Yield	4%	3%	
Maturity	3	6	3.6
Rating	A = 2	Aaa = 1	1.5
Allocations	??	??	100,000

We can model the above problem as follows:

$$\begin{aligned}
 &\text{maximize: } 4x + 3y \\
 &\text{subject to: } 3x + 6y \leq 3.6 \\
 &\quad \quad \quad 2x + y \leq 1.5
 \end{aligned} \tag{1.1}$$

where x, y are the percentages of funds allocated to corporate and government bonds, respectively, and the objective function when multiplied by \$100,000 gives us the net yield. This is an example of a linear program. Note that we cannot subtract inequalities the same way that we subtract equations. So to get started, let us assume that both the inequalities are in fact equations. We can solve the resulting system,

$$\begin{aligned}
 3x + 6y &= 3.6 \\
 2x + y &= 1.5,
 \end{aligned}$$

to obtain $x = 0.6$ and $y = 0.3$. For this solution $x + y = 0.9$ which is less than 1, meaning that we're not investing all the available funds. This seems less than ideal! Surely, we must invest all the funds to get maximum yield!!

To complicate things further, notice that the linear program (1.1) is incomplete and the complete linear program that models the problem is as follows:

$$\begin{aligned}
 &\text{maximize: } 4x + 3y \\
 &\text{subject to: } 3x + 6y \leq 3.6 \\
 &\quad \quad \quad 2x + y \leq 1.5 \\
 &\quad \quad \quad x + y \leq 1 \\
 &\quad \quad \quad x \geq 0 \\
 &\quad \quad \quad y \geq 0.
 \end{aligned} \tag{1.2}$$

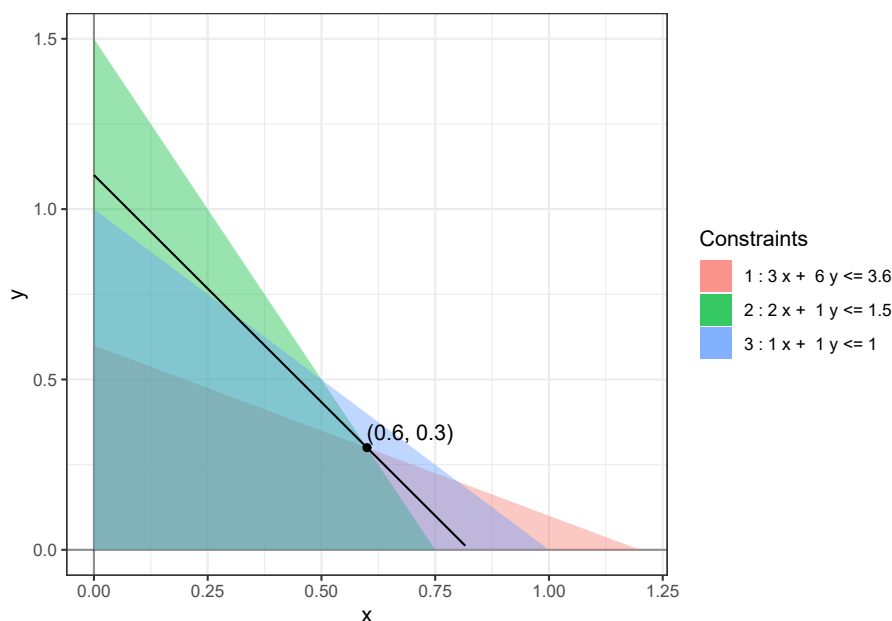
The solution $(x, y) = (0.6, 0.3)$ is obtained by changing the first two inequalities to equations. We could have switched some other set of inequalities to equations and solved for a different solution. We would then need to compare the yields for each of these solutions and find the one that maximizes it.

However, now there is an additional issue that needs to be dealt with carefully. Solving these modified equations might produce a solution that does not satisfy

the original constraints. For example, solving the system of equations

$$\begin{array}{rcl} 2x + y & = & 1.5 \\ x & = & 0 \end{array}$$

gives us $(x, y) = (0, 1.5)$ which breaks the constraints $3x + 6y \leq 3.6$ and $x + y \leq 1$. So, not only do we need to compare the yields at these solutions but we also need to make sure that they satisfy all the original constraints. As you can imagine, this naive approach becomes unwieldy for large linear programs. It is not even clear that changing inequalities to equations is the right approach to solving such a problem.¹ For this particular problem, we can use a graphing calculator to solve the problem as there are only two variables. The following figure shows the region described by the various constraints in the first quadrant ($x \geq 0, y \geq 0$).



The points within the quadrilateral formed by the overlap of all three constraint regions satisfy all the constraints. The level sets of the objective function are straight lines of the form $4x + 3y = c$. We can see that the largest value of c for which these level sets intersect the feasible region is $c = 3.3\%$ for $(x, y) = (0.6, 0.3)$. Our original solution was indeed correct! And it is not possible to invest more money and increase the yield!!

Exercise 1.1. What are some possible ways to relax the constraints in the above problem so that all the money can be invested? What is the improved annual yield?

¹That this is indeed the case is the content of the Fundamental Theorem of Linear Programming (Theorem 2.1).

Exercise 1.2. The following exercises review some basic facts about inequalities:

1. Find constants a, b, c, d such that $a < b$, $c < d$, $a - c < b - d$, and $ac < bd$.
2. Find constants a, b, c, d such that $a < b$, $c < d$, $a - c < b - d$, and $ac > bd$.
3. Find constants a, b, c, d such that $a < b$, $c < d$, $a - c > b - d$, and $ac < bd$.
4. Find constants a, b, c, d such that $a < b$, $c < d$, $a - c > b - d$, and $ac > bd$.
5. Prove that if a, b, c, d are constants such that $a < b$ and $c < d$, then $a + c < b + d$. Why does this proof fail when we try to subtract the two inequalities instead of adding them?

There are two main families of efficient algorithms for solving linear programs: simplex method(s) (Chapter 3) and interior point methods (Chapter 12). The simplex method is a combinatorial algorithm that essentially performs the above analysis systematically. Interior point methods modify gradient descent algorithms to search for an optimal solution within the feasible region. We'll analyze the simplex method in great detail and only briefly introduce a basic interior point method.

Part I

Simplex Method

Chapter 2

Standard Linear Program

We will describe the simplex method for solving a special kind of linear program called standard linear program. We will see later that every linear program can be standardized (Chapter 6) and hence this method is sufficient for solving any linear program.

Definition 2.1. A standard linear program is an optimization problem of the following form:

$$\begin{array}{llllllll}
 \text{maximize:} & c_0 & + & c_1 x_1 & + & \dots & + & c_n x_n \\
 \text{subject to:} & & & a_{11} x_1 & + & \dots & + & a_{1n} x_n & \leq & b_1 \\
 & & & a_{21} x_1 & + & \dots & + & a_{2n} x_n & \leq & b_2 \\
 & & & & & \vdots & & & & \\
 & & & a_{m1} x_1 & + & \dots & + & a_{mn} x_n & \leq & b_m \\
 & & & x_1, & x_2, & \dots & , & x_n & \geq & 0
 \end{array} \tag{2.1}$$

where c_i , a_{ij} , and b_j are real constants. The variables x_1, \dots, x_n are called decision variables. A tuple (x_1, \dots, x_n) that satisfy all the constraints is called a feasible solution and the set of all feasible solutions is called the feasible region.

Example 2.1. Equation (1.2) is an example of a standard linear program with 2 decision variables and 3 constraints. The feasible region is a quadrilateral with vertices $(0, 0)$, $(0.75, 0)$, $(0.6, 0.3)$, and $(0, 0.6)$.

Definition 2.2. For each constraint of the standard linear program (2.1), we introduce a slack variable by subtracting the LHS from the RHS as follows.

$$\begin{array}{llllllll}
 w_1 & = & b_1 & - & a_{11} x_1 & - & \dots & - & a_{1n} x_n \\
 w_2 & = & b_2 & - & a_{21} x_1 & - & \dots & - & a_{2n} x_n \\
 & & & & & & \vdots & & \\
 w_m & = & b_m & - & a_{m1} x_1 & - & \dots & - & a_{mn} x_n
 \end{array} \tag{2.2}$$

We can think of the slack variable w_i as measuring the slackness in the i^{th} constraint. The i^{th} constraint is strictly met exactly when w_i is zero. Using the

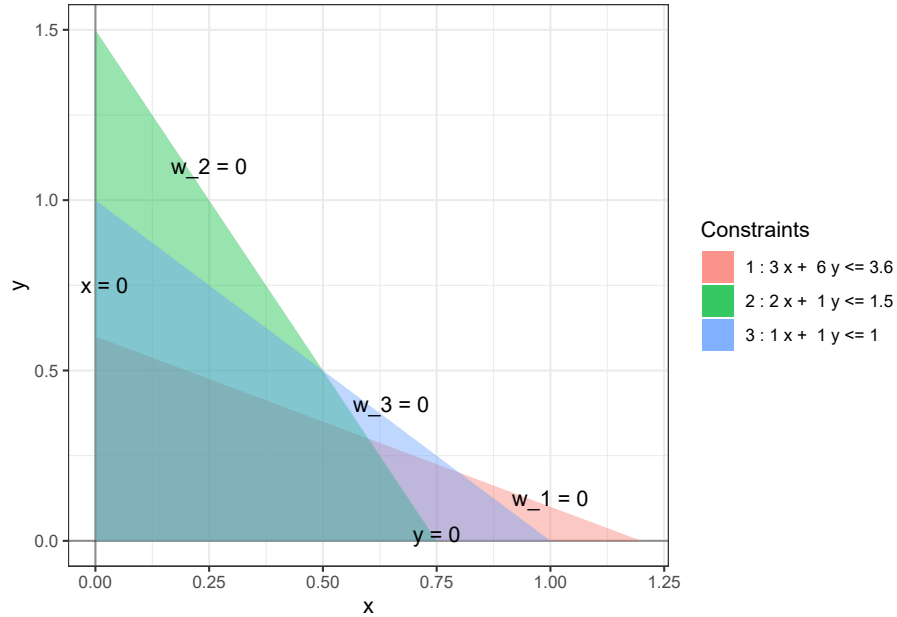
slack variables, the linear program (2.1) can be succinctly rewritten as:

$$w_1, \dots, w_m, x_1, \dots, x_n \geq 0.$$

Example 2.2. The slack variables for the linear program (1.2) are as follows:

$$\begin{aligned} w_1 &= 3.6 - 3x - 6y \\ w_2 &= 1.5 - 2x - y \\ w_3 &= 1 - x - y. \end{aligned}$$

In terms of these slack variables, the constraints can be rewritten as $x, y, w_1, w_2, w_3 \geq 0$ and the boundaries of the feasible region are given by $x = 0, y = 0, w_1 = 0, w_2 = 0, w_3 = 0$.



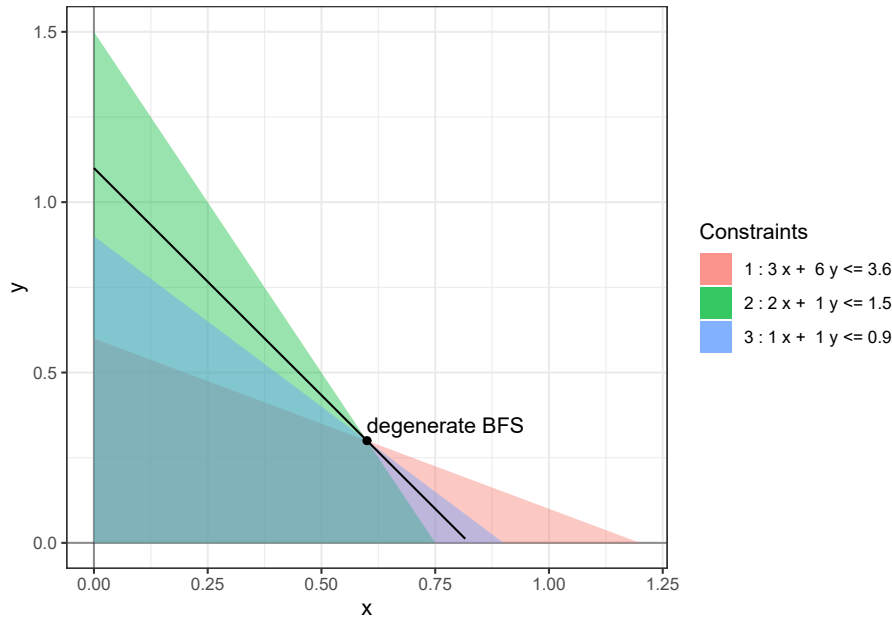
Definition 2.3. A basic feasible solution (BFS) to the standard linear program (2.1) is defined as a feasible solution at which at least n decision or slack variables are zero. A BFS where exactly n decision or slack variables are zero is called non-degenerate. A BFS where more than n basic or decision variables are 0 is called degenerate. At a non-degenerate BFS, the n variables that equal 0 are called non-basic and the remaining m variables are called basic. At a degenerate BFS, we choose some n vanishing variables to be non-basic and call the rest of the variables basic.

The basic feasible solutions are precisely the solutions obtained by changing n or more inequalities to equations and solving the resulting simultaneous system of equations. As there are n variables, we need at least n linear equations in our system for the solution to be a single point. A degeneracy occurs when more

than n equations have a single solution, for example, when three lines in \mathbb{R}^2 or four planes in \mathbb{R}^3 intersecting at a single point. It is useful to think of the non-degenerate basic feasible solutions as vertices of the feasible region. The degenerate BFS do not have a nice geometric description.

Exercise 2.1. Show that if $k < n$ then the solution set of a system of k linear equations in n variables cannot be a single point.

Example 2.3. The feasible region for the linear program (1.2) has four non-degenerate basic feasible solutions: $(0, 0)$, $(0.75, 0)$, $(0.6, 0.3)$, and $(0, 0.6)$. and the optimal solution is attained at the BFS $(0.6, 0.3)$. At the origin, the non-basic variables are x, y and the basic variables are w_1, w_2, w_3 . At the optimal solution, the non-basic variables are w_1, w_2 and the basic variables are x, y, w_3 . If we replace the constraint $x + y \leq 1$ with $x + y \leq 0.9$ then the BFS $(0.6, 0.3)$ becomes degenerate as here three (slack) variables w_1, w_2, w_3 are zero.



It turns out that it is sufficient to search for optimal solutions within the set of basic feasible solutions, which is finite, as made precise by the following theorem.

Theorem 2.1 (Fundamental theorem of linear programming). For a standard linear program, exactly one of the following holds:

1. There is no feasible solution. In this case, we call the linear program infeasible.
2. The objective value can grow arbitrary large on the feasible region. In this case, we call the linear program unbounded.
3. There is an optimal solution. In this case, we can further say that then

there is a basic feasible solution which is optimal.

We will assume this theorem without proof. The hardest step in the proof is showing that some basic feasible solution is optimal (if an optimal solution exists) when there are degenerate basic feasible solutions.

Exercise 2.2. Find an upper bound on the set of basic feasible solutions for a standard linear program with n decision variables and m equations.

Chapter 3

The Simplex Method

The simplex method is an iterative process for finding an optimal basic feasible solution to a standard linear program. It starts at some BFS and in each step moves to an adjacent one with a higher objective value. The following picture shows one possible run of the simplex method for the linear program (1.2).

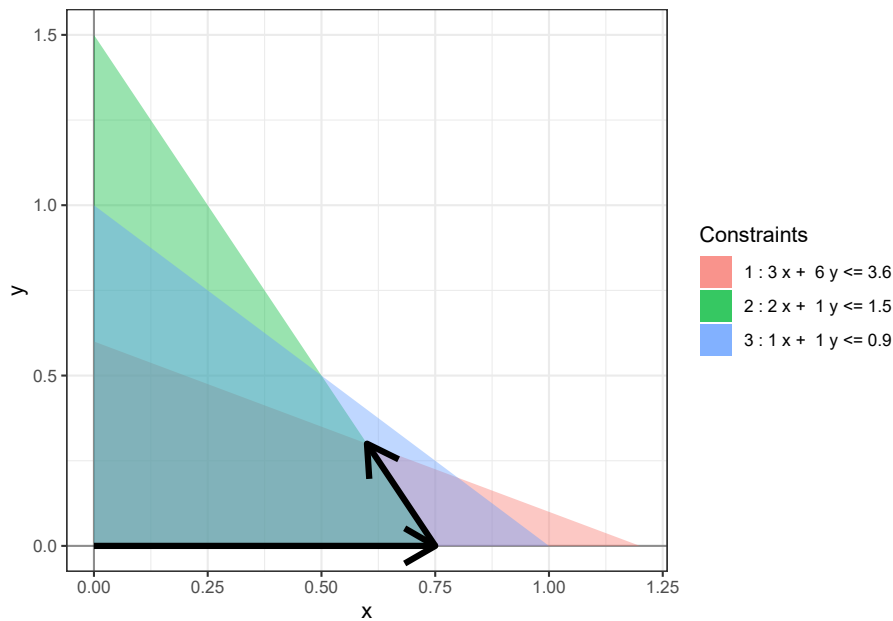
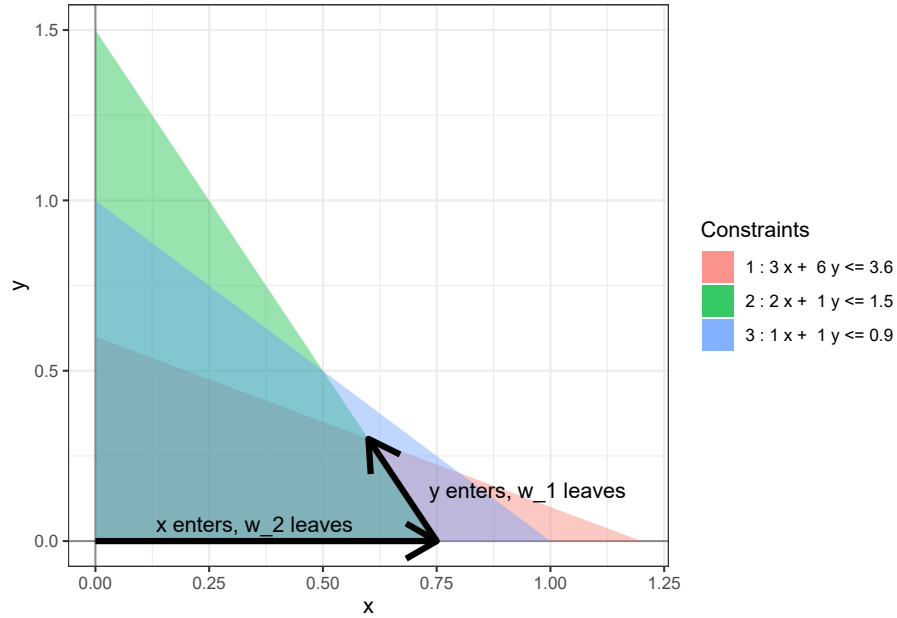


Figure 3.1: A possible run of the simplex algorithm.

In each step, one non-basic variable enters the set of basic variables and one basic variable leaves the set of basic variables. The table below explains how

these sets are getting updated in the sample simplex method run in Figure 3.1.

	Leaving variable	Entering variable	Basic variables	Non-basic variables
Start			$\{w_1, w_2, w_3\}$	$\{x, y\}$
Step 1	w_2	x	$\{w_1, x, w_3\}$	$\{w_2, y\}$
Step 2	w_1	y	$\{y, x, w_3\}$	$\{w_2, w_1\}$



Using this terminology, our goal at each step is reduced to finding which variable enters and which variable leaves. To do this algebraically, we'll introduce the notion of dictionaries.

Definition 3.1. The dictionary at a basic feasible solution is the set of equations describing the objective function and the constraints in terms of the non-basic variables.

Example 3.1. Consider (1.2) again. At the origin, the non-basic variables are x, y and hence the initial dictionary is:

$$\begin{aligned}
 \text{objective} &= 0 + 4x + 3y \\
 w_1 &= 3.6 - 3x - 6y \\
 w_2 &= 1.5 - 2x - y \\
 w_3 &= 1 - x - y.
 \end{aligned} \tag{3.1}$$

After the first step of the simplex algorithm, the non-basic variables are w_2, y .

We can write x in terms of w_2 to get

$$x = 0.75 - 0.5w_2 - 0.5y$$

We can substitute this into the initial dictionary to get the dictionary after the first step:

$$\begin{aligned} \text{objective} &= 3 + (-2)w_2 + y \\ w_1 &= 1.35 - (-1.5)w_2 - 4.5y \\ x &= 0.75 - 0.5w_2 - 0.5y \\ w_3 &= 0.25 - (-0.5)w_2 - 0.5y. \end{aligned} \tag{3.2}$$

Finally, the non-basic variables at the optimal solution are w_1, w_2 . We can repeat the above process and get the dictionary for the optimal solution:

$$\begin{aligned} \text{objective} &= 3.3 + (-5/3)w_2 + (-2/9)w_1 \\ y &= 0.3 - (-1/3)w_2 - 2/9w_1 \\ x &= 0.6 - 2/3w_2 - (-1/9)w_1 \\ w_3 &= 0.1 - (-1/3)w_2 - (-1/9)w_1. \end{aligned} \tag{3.3}$$

Remark. From the dictionary, one can extract the set of basic variables and the set of non-basic variables by looking at the variables appearing on the LHS and RHS, respectively. Furthermore, by setting the non-basic variables to 0, we obtain the very useful fact that

1. the values of the basic variables are simply the constants " b_i ", and
2. the value of the objective function is the constant " c_0 ".

For example, from the final dictionary above, we can immediately see that $x = 0.6$, $y = 0.3$, $w_3 = 0.1$, and the objective value is 3.3 (and non-basic variables w_1 and w_2 are both zero) at the optimal solution.

3.1 The Simplex Step

Suppose we are at a BFS. Let $\bar{w}_1, \dots, \bar{w}_m$ be the basic variables, $\bar{x}_1, \dots, \bar{x}_n$ be the non-basic variables, \bar{c}_j , \bar{b}_i , and \bar{a}_{ij} be the constants appearing in the dictionary at the BFS so that the dictionary is as follows:

$$\begin{aligned} \text{objective} &= \bar{c}_0 + \bar{c}_1\bar{x}_1 + \dots + \bar{c}_n\bar{x}_n \\ \bar{w}_1 &= \bar{b}_1 - \bar{a}_{11}\bar{x}_1 - \dots - \bar{a}_{1n}\bar{x}_n \\ \bar{w}_2 &= \bar{b}_2 - \bar{a}_{21}\bar{x}_1 - \dots - \bar{a}_{2n}\bar{x}_n \\ &\vdots \\ \bar{w}_m &= \bar{b}_m - \bar{a}_{m1}\bar{x}_1 - \dots - \bar{a}_{mn}\bar{x}_n \end{aligned} \tag{3.4}$$

3.1.1 Entering variable

We want to move to an adjacent BFS with a higher objective value. The current objective function is

$$\bar{c}_0 + \bar{c}_1\bar{x}_1 + \dots + \bar{c}_n\bar{x}_n.$$

and the current objective value is \bar{c}_0 as all the non-basic variables \bar{x}_j are zero. We can move away from the current BFS by increasing one of the non-basic variables \bar{x}_j from zero to a positive value. The objective value will increase precisely when $\bar{c}_j > 0$. This then is the criterion for choosing the entering variable. We can think of this as choosing the direction of the simplex step.

Proposition 3.1 (Entering variable). A non-basic variable \bar{x}_j can be entering if $\bar{c}_j > 0$.

Example 3.2. In the dictionary (3.1), the objective function is $4x + 3y$. Hence, both x and y can be chosen as entering variables. We can see that there are two different paths along the boundary of the feasible region from the origin to the optimal solution. In the dictionary (3.2), the objective function is $3 + (-2)w_2 + y$ and only y can be the entering variable. In dictionary (3.3), the objective function is $3.3 + (-5/3)w_2 + (-2/9)w_1$ and no variable can enter.

3.1.2 Leaving Variable

Suppose we choose \bar{x}_j as our entering variable. This determines the direction of the simplex step. We next need to figure out the amount by which to increase \bar{x}_j without leaving the feasible region. The basic variable \bar{w}_i is related to \bar{x}_j by the following relation

$$\bar{w}_i = \bar{b}_i - \bar{a}_{ij}\bar{x}_j - \sum_{k=1, k \neq j}^n \bar{a}_{ik}\bar{x}_k.$$

As we increase (only) \bar{x}_j , the quantity within the summation remains zero. If $\bar{a}_{ij} > 0$, then as we increase \bar{x}_j the basic variable \bar{w}_i will decrease. Because we want all the variables to be non-negative, we must always have $\bar{w}_i = \bar{b}_i - \bar{a}_{ij}\bar{x}_j \geq 0$. This condition must hold true for all such \bar{w}_i . Hence, we get the following criterion for choosing the leaving variable.

Proposition 3.2 (Leaving variable). Suppose \bar{x}_j is the entering variable. The basic variable \bar{w}_i can be chosen to be the leaving variable if

$$i = \arg \min_{\bar{a}_{ij} > 0} \frac{\bar{b}_i}{\bar{a}_{ij}}.$$

If we choose \bar{w}_i to be leaving then after the simplex step, \bar{w}_i decreases to 0, \bar{x}_j increases to $\min_{\bar{a}_{ij} > 0} \bar{b}_i/\bar{a}_{ij}$, and the objective value increases by $\bar{c}_j \min_{\bar{a}_{ij} > 0} \bar{b}_i/\bar{a}_{ij}$.

Example 3.3. In the dictionary (3.1), if we choose x to be our entering variable then we need to compare the following ratios:

i	\bar{a}_{ij}	\bar{b}_i	\bar{b}_i/\bar{a}_{ij}
1	3	3.6	1.2

i	\bar{a}_{ij}	\bar{b}_i	\bar{b}_i/\bar{a}_{ij}
2	2	1.5	0.75
3	1	1	1

We can see that the smallest ratio is obtained for w_2 . Hence, it is the only candidate for the leaving variable.

3.2 Tableau Notation

When manipulating a linear system of equations, one can forget the variables and perform manipulations on the coefficients using matrices. The same is true for linear programs. We start by rewriting the constraints in the dictionary with all the variables on the LHS and all the constants on the RHS:

$$\begin{array}{ccccccc}
\bar{a}_{11}\bar{x}_1 & + & \dots & + & \bar{a}_{1n}\bar{x}_n & + & \bar{w}_1 & = & \bar{b}_1 \\
\bar{a}_{21}\bar{x}_1 & + & \dots & + & \bar{a}_{2n}\bar{x}_n & & + & \bar{w}_2 & = & \bar{b}_2 \\
& & & & & & \vdots & & & \\
\bar{a}_{m1}\bar{x}_1 & + & \dots & + & \bar{a}_{mn}\bar{x}_n & & & + & \bar{w}_m & = & \bar{b}_m
\end{array} \quad (3.5)$$

This can be encoded using the following augmented matrix:

$$\begin{array}{cccc|c}
\bar{a}_{11} & \dots & \bar{a}_{1n} & 1 & \bar{b}_1 \\
\bar{a}_{21} & \dots & \bar{a}_{2n} & & \bar{b}_2 \\
& & \vdots & & \\
\bar{a}_{m1} & \dots & \bar{a}_{mn} & & \bar{b}_m
\end{array}$$

We add back the objective function, but because of a quirk of algebra we need to add the objective function coefficients as follows:

$$\begin{array}{cccc|c}
\bar{a}_{11} & \dots & \bar{a}_{1n} & 1 & \bar{b}_1 \\
\bar{a}_{21} & \dots & \bar{a}_{2n} & & \bar{b}_2 \\
& & \vdots & & \\
\bar{a}_{m1} & \dots & \bar{a}_{mn} & & \bar{b}_m \\
\hline
\bar{c}_1 & \dots & \bar{c}_n & 0 & 0 \dots 0 & -\bar{c}_0
\end{array}$$

The columns in this augmented matrix correspond to the variables \bar{x}_j and \bar{w}_i . The columns with the pivots correspond to the basic variables. If \bar{x}_j is the entering variable and \bar{w}_i is the leaving variable, then we perform elementary row operations and turn the entry \bar{a}_{ij} into a pivot for its column to obtain the tableau at the next BFS. Hence, the simplex step is also called the pivot step.

Example 3.4. The tableau corresponding to the dictionary (3.1) is as follows:

$$\begin{array}{cccc|c}
3 & 6 & 1 & 0 & 0 & 3.6 \\
\boxed{2} & 1 & 0 & 1 & 0 & 1.5 \\
1 & 1 & 0 & 0 & 1 & 1 \\
\hline
4 & 3 & 0 & 0 & 0 & 0
\end{array}$$

If we choose x as the entering variable and w_2 as the leaving variable then we need to pivot about the entry a_{21} using elementary row operations to get the following tableau:

$$\begin{array}{ccccc|c}
 0 & 4.5 & 1 & -1.5 & 0 & 1.35 \\
 \boxed{1} & 0.5 & 0 & 0.5 & 0 & 0.75 \\
 0 & 0.5 & 0 & -0.5 & 1 & 0.25 \\
 \hline
 0 & 1 & 0 & -2 & 0 & -3
 \end{array}$$

which corresponds to the dictionary (3.2).

Chapter 4

Initialization

So far, we've seen how to move from one basic feasible solution to an adjacent one with a higher objective value. To have a complete algorithm for solving standard linear programs, we still need a method that finds some BFS. If we're lucky, the origin is a basic feasible solution and we can start our simplex method here. However, this is not always the case. The following proposition follows easily from definitions.

Proposition 4.1. The origin is a basic feasible solution of the standard linear program (2.1) if and only if $b_i \geq 0$ for all $1 \leq i \leq m$.

When the origin is not a BFS, we need a process to find one. This is called the Initialization Phase or Phase I. We will do this by creating an auxiliary linear program.

4.1 Auxiliary Linear Program

Definition 4.1. We say that a linear program is feasible if its feasible region is non-empty.

The initialization phase determines if a standard linear program is feasible and if it is then it finds a BFS by solving the following auxiliary linear program.

Definition 4.2. The auxiliary linear program of the standard linear program (2.1) is defined as follows:

$$\begin{array}{llllllllll} \text{maximize:} & & & & & & - & x_0 & & & \\ \text{subject to:} & a_{11}x_1 & + & \dots & + & a_{1n}x_n & - & x_0 & \leq & b_1 & \\ & a_{21}x_1 & + & \dots & + & a_{2n}x_n & - & x_0 & \leq & b_2 & \\ & & & \vdots & & & & & & & \\ & a_{m1}x_1 & + & \dots & + & a_{mn}x_n & - & x_0 & \leq & b_m & \\ & x_1, & x_2, & \dots & , & x_n & , & x_0 & \geq & 0. & \end{array} \quad (4.1)$$

We're introducing a new variable $-x_0$, adding $-x_0$ to each of the constraint LHS, and changing the objective function to $-x_0$. To understand the auxiliary linear program, we rewrite it in the following non-standard form:

$$\begin{array}{ll}
 \text{minimize:} & x_0 \\
 \text{subject to:} & a_{11}x_1 + \dots + a_{1n}x_n \leq b_1 + x_0 \\
 & a_{21}x_1 + \dots + a_{2n}x_n \leq b_2 + x_0 \\
 & \vdots \\
 & a_{m1}x_1 + \dots + a_{mn}x_n \leq b_m + x_0 \\
 & x_1, x_2, \dots, x_n, x_0 \geq 0
 \end{array}$$

We can interpret x_0 as a relaxation of the constraints. Solving the auxiliary linear program is equivalent to asking - "what is the smallest constraint relaxation necessary to make the primary linear program feasible?". The primary linear program is feasible if and only if no relaxation is necessary. The following exercises make this more precise.

Exercise 4.1. Show that the auxiliary linear program (4.1) is always feasible by explicitly constructing a feasible solution.

Exercise 4.2. Show that the objective values of the auxiliary linear program (4.1) are bounded from above by 0. Using the extreme value theorem, conclude that (4.1) always has an optimal solution.

Finally, the following theorem (proof left as an exercise) establishes an explicit connection between optimal solutions of the auxiliary linear program and feasible solutions of the standard linear program.

Theorem 4.1. Suppose $x_i = k_i$ for $0 \leq i \leq m$ is an optimal solution of the auxiliary linear program (4.1). Then, the standard linear program (2.1) is feasible if and only if $k_0 = 0$. In this case, $x_i = k_i$ for $0 \leq i \leq m$ is a basic feasible solution of (2.1).

Our problem is now reduced to determining the optimal solution of the auxiliary linear program, which we'll do using the simplex method itself! However, even for the auxiliary linear program the origin is not necessarily in the feasible region so we cannot use the simplex steps to find an optimal solution. But, in this case, there is a trick to make all the $b_i \geq 0$ by performing a single elementary row operation as described in the following exercises.

Exercise 4.3. Suppose some b_i is negative. Without any loss of generality, assume that b_1 is the smallest among all the b_i (i.e. the most negative). Show that after the pivot step in tableau for the auxiliary linear program (4.1) about the column corresponding to x_0 and the row corresponding to w_1 , all the b_i 's become positive.

To summarize: If the origin is not a vertex of the feasible region, then the method of solving the standard linear program (2.1), starting with Phase I, is as follows:

1. Form the auxiliary linear program and its tableau.
2. Perform a pivot operation about the entry in the column corresponding to the variable x_0 and the row corresponding to the most negative b_i . This results in a dictionary where all the b_i 's are now non-negative.
3. Solve the auxiliary linear program by repeatedly performing pivot steps. Suppose $x_i = k_i$ for $0 \leq i \leq m$ is an optimal solution of the auxiliary linear program.
 1. If $k_0 \neq 0$, then we halt as the primary linear program is not feasible.
 2. If $k_0 = 0$, then we proceed to Phase II. We solve the original linear program starting at the BFS $x_i = k_i$ for $1 \leq i \leq n$ by repeatedly performing pivot steps.

Exercise 4.4. This exercise provides a geometric intuition behind the auxiliary linear program. For each of the following standard linear programs,

$$\begin{array}{ll} \text{maximize:} & x \\ \text{subject to:} & -x \leq -1 \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \text{maximize:} & x \\ \text{subject to:} & x \leq -1 \\ & x \geq 0 \end{array}$$

- Form the auxiliary linear program. Use y as the auxiliary variable.
- Draw the feasible region of the auxiliary linear program and label the constraints using slack variables.
- Solve the auxiliary linear program using geometry.

4.2 Combined tableau

At the end of Phase I, we need to calculate the dictionary at the basic feasible solution $x_i = k_i$ for $1 \leq i \leq n$. We can avoid this recalculation by manipulating a combined tableau which contains information about both the auxiliary and the primary linear programs.

Definition 4.3. The combined tableau is defined as follows:

$$\begin{array}{cccccccc|c} a_{11} & \dots & a_{1n} & 1 & & & & -1 & b_1 \\ a_{21} & \dots & a_{2n} & & 1 & & & -1 & b_2 \\ & & & \vdots & & & & \vdots & \vdots \\ a_{m1} & \dots & a_{mn} & & & & 1 & -1 & b_m \\ \hline c_1 & \dots & c_n & 0 & 0 & \dots & 0 & 0 & -c_0 \\ \hline 0 & \dots & 0 & 0 & 0 & \dots & 0 & -1 & 0 \end{array}$$

The second to last row of the tableau corresponds to the objective function of the primary linear program and the last row corresponds to the objective of the

auxiliary linear program. The second to last column corresponds to the variable x_0 .

We use the combined tableau to first perform Phase I and then delete the auxiliary objective and the variable x_0 and proceed on to Phase II using the rest of the tableau.

Example 4.1. Consider the following linear program:

$$\begin{array}{llllll} \text{maximize:} & x & + & y & & \\ \text{subject to:} & x & + & 2y & \leq & 6 \\ & -x & & & \leq & -1 \\ & & & -y & \leq & -2 \\ & x & , & y & \geq & 0. \end{array}$$

As b_2 and b_3 are negative, $(0, 0)$ is not feasible and we need to start with Phase I. We form the combined tableau:

$$\begin{array}{cccccc|c} 1 & 2 & 1 & 0 & 0 & -1 & 6 \\ -1 & 0 & 0 & 1 & 0 & -1 & -1 \\ 0 & -1 & 0 & 0 & 1 & \boxed{-1} & -2 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{array}$$

The first pivot is in the column corresponding to the variable x_0 and the row corresponding to the most negative b_i (which is $b_3 = -2$). This results in the following combined tableau:

$$\begin{array}{cccccc|c} 1 & 3 & 1 & 0 & -1 & 0 & 8 \\ -1 & 1 & 0 & 1 & -1 & 0 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 & 2 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & -1 & 0 & 2 \end{array}$$

We then continue performing pivot steps to find the solution:

$$\begin{array}{cccccc|c} 0 & 0 & 1 & 1 & 2 & -4 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 & 2 \\ 1 & 0 & 0 & -1 & 0 & 1 & 1 \\ \hline 0 & 0 & 0 & 1 & 1 & -2 & -3 \\ \hline 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{array}$$

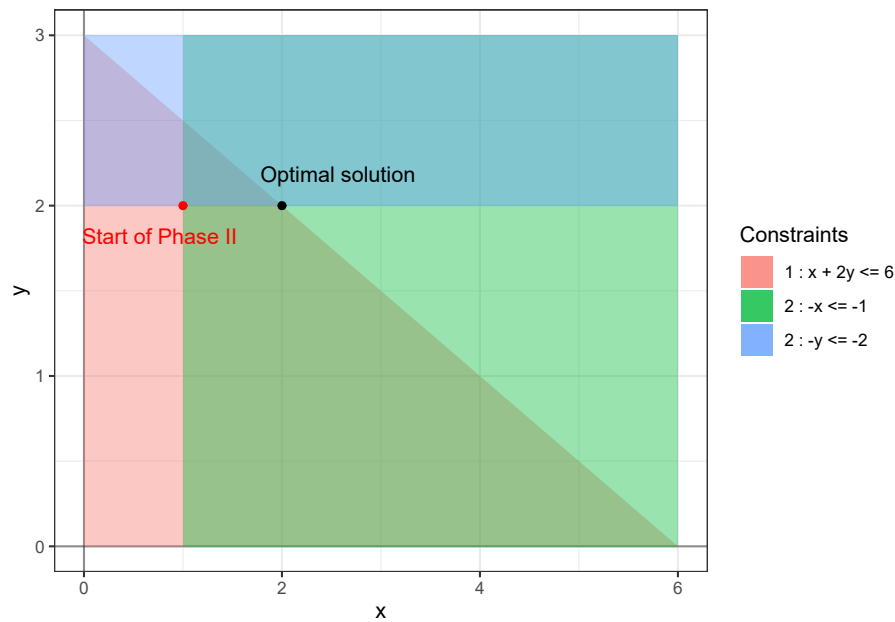
We see that the optimal value is 0 and hence the primary linear program is feasible. We then remove the auxiliary objective and coefficient to get the following tableau for the primary linear program:

$$\begin{array}{cccc|c} 0 & 0 & 1 & \boxed{1} & 2 & 1 \\ 0 & 1 & 0 & 0 & -1 & 2 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & 1 & -3 \end{array}$$

Continuing with the simplex method we get the following final tableau

0	0	1	1	2	1
0	1	0	0	-1	2
1	0	1	0	2	2
0	0	-1	0	-1	-4

The final non-basic variables are w_1 and w_3 , the basic variables are x , y , w_2 with values 2, 2, 1, respectively, and the optimal objective value is 4. The following figure shows the starting and ending basic feasible solutions for Phase II.



Chapter 5

Cycling

We know how to start the simplex method and how to perform the pivot steps. We'll next analyze the halting conditions for the simplex method. We will not be able to perform the pivot steps if no entering or leaving variables are found.

If no entering variable is found, then the geometry tells us that there is no direction in which the objective value can be increased i.e. we're at a local maxima. But because the objective function is a linear function this local maxima is also an absolute maxima and provides an optimal solution to our linear program. Algebraically, this happens when none of the \bar{c}_i are positive.

Proposition 5.1. If none of the objective coefficients \bar{c}_i are positive in the current dictionary, then the current BFS is optimal.

If no leaving variable is found, then the geometry tells us that we can keep increasing the entering variable indefinitely without leaving the feasible region. Such a linear program is called unbounded. An unbounded linear program has no optimal solution as the objective value can be made arbitrary large without leaving the feasible region. Algebraically, this happens when none of the \bar{a}_{ij} are positive.

Proposition 5.2. Suppose \bar{x}_j is the entering variable. If none of the constants \bar{a}_{ij} are positive in the current dictionary, then the linear program is unbounded.

Another point of failure is in the initialization phase. We've already seen that if the initialization phase fails then there is no feasible solution. We restate the result here for completion.

Proposition 5.3. If some b_i is negative, and the initialization phase fails then the linear program is infeasible.

Unfortunately, the above results are not sufficient to guarantee that the simplex method will always find a solution if one exists. It is possible for the simplex method to get stuck in a loop. This is called cycling.

Exercise 5.1. Give an example showing that the variable that enters in one pivot step can become leave in the next.

Exercise 5.2. Show that the variable that leaves in one pivot step cannot enter in the next.

5.1 Degeneracy

Cycling does not always occur when running the simplex method. We saw in Section 3.1 that after the pivot step the entering and leaving variable gets updated as follows:

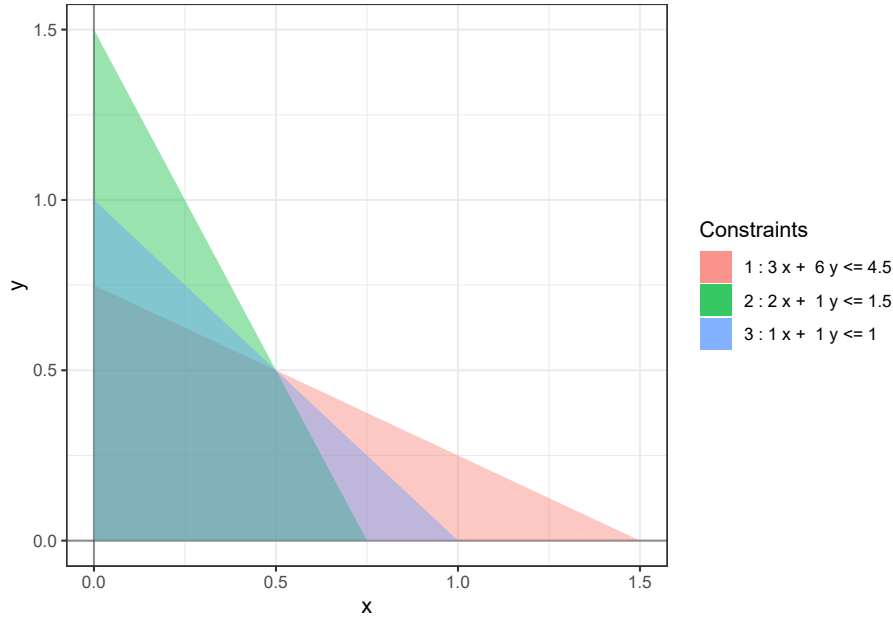
$$\begin{aligned}\bar{x}_j &\mapsto \bar{b}_i / \bar{a}_{ij} \\ \bar{w}_i &\mapsto 0.\end{aligned}$$

This increases the value of the objective function by $\bar{c}_j \bar{b}_i / \bar{a}_{ij}$. Because of the criterion for choosing the entering and leaving variables, the constants \bar{c}_i and \bar{a}_{ij} are always positive. We know that \bar{b}_i this is the value of the basic variable \bar{w}_i and hence must be greater than or equal to 0. If $\bar{b}_i > 0$ then the objective value will increase in the pivot step and we will never return back to this BFS. Hence, the only case when cycling can occur is when $\bar{b}_i = 0$. But \bar{b}_i is the value of the basic variable \bar{w}_i . Having more than n variables vanishing at a BFS is precisely the definition of degeneracy 2.3.

Proposition 5.4. The simplex method can cycle if there some degenerate BFS.

Example 5.1. The following slight modification of Example (1.2) is a degenerate linear program:

$$\begin{array}{llllll} \text{maximize:} & 4x & + & 3y & & \\ \text{subject to:} & 3x & + & 6y & \leq & 4.5 \\ & 2x & + & y & \leq & 1.5 \\ & x & + & y & \leq & 1 \\ & x & , & y & \geq & 0. \end{array}$$



At the optimal solution, $(0.5, 0.5)$ all three constraints are met. At this BFS, one of the rows is

$$w_3 = 0 + 0.33w_2 + 0.11w_1. \quad (5.1)$$

Example 5.2. Consider the following degenerate linear program:

$$\begin{array}{llllll} \text{maximize:} & x_1 & - & 2x_2 & & - & 2x_4 \\ \text{subject to:} & 0.5x_1 & - & 3.5x_2 & - & 2x_3 & + & 4x_4 & \leq & 0 \\ & 0.5x_1 & - & x_2 & - & 0.5x_3 & + & 0.5x_4 & \leq & 0 \\ & x_1 & & & & & & & \leq & 1 \\ & x_1 & , & x_2 & , & x_3 & , & x_4 & \geq & 0 \end{array}$$

The following is a valid sequence of simplex steps:

1. x_1 enters and w_1 leaves,
2. x_2 enters and w_2 leaves,
3. x_3 enters and x_1 leaves,
4. x_4 enters and x_2 leaves,
5. w_1 enters and x_3 leaves,
6. w_2 enters and x_4 leaves.

At the end of the 6th simplex step, we end up looping back to the origin.

5.2 Bland's Rule

There are various ways of dealing with cycling. The simplest such way is called Bland's rule. Bland's rule says that if there are multiple candidates for the entering variable then we choose the one with the smallest index. (We assume that the decision variables have a smaller index than the slack variables.)

Theorem 5.1 (Bland's rule). The simplex method always terminates provided that both the entering and the leaving variable are chosen according to Bland's rule.

The proof of this theorem is too complicated for this course. With this modification, for both Phase I and Phase II of the simplex method, we now have a complete algorithm for solving linear programs.

Example 5.3. In Example 5.2, the sixth simplex step violates Bland's rule. Both x_1 and x_4 can be leaving variables and we choose x_4 whereas Bland's rule requires us to choose x_1 .

Chapter 6

Standardization

We have an algorithm for solving a standard linear program. We'll not extend it to more general linear programs, which is simply an optimization problem where the objective function is a linear program and the constraints are linear equalities or inequalities.

Definition 6.1. A general linear program is an optimization problem of the following form:

$$\begin{array}{llllll}
 \text{optimize:} & c_1x_1 & + & \dots & + & c_nx_n \\
 \text{subject to:} & a_{11}x_1 & + & \dots & + & a_{1n}x_n & \begin{array}{l} \leq \\ = \\ \geq \end{array} & b_1 \\
 & a_{21}x_1 & + & \dots & + & a_{2n}x_n & \begin{array}{l} \leq \\ = \\ \geq \end{array} & b_2 \\
 & & & \vdots & & & & \\
 & a_{m1}x_1 & + & \dots & + & a_{mn}x_n & \begin{array}{l} \leq \\ = \\ \geq \end{array} & b_m,
 \end{array} \tag{6.1}$$

where the symbol $\begin{array}{l} \leq \\ = \\ \geq \end{array}$ stands for \leq or $=$ or \geq , a_{ij}, b_i, c_j are real numbers, and x_j are variables.

We do not allow strict inequalities $<$ or $>$ in a linear program as linear functions do not always achieve maxima/minima on open sets even when the sets are bounded. Consider the following simple example.

$$\begin{array}{ll}
 \text{maximize:} & x \\
 \text{subject to:} & x < 1 \\
 & x \geq 0
 \end{array}$$

On the feasible set $[0, 1)$, the function x never attains absolute maxima. Changing the inequality $<$ to \leq gives us an optimal feasible solution $x = 1$.

6.1 Equivalence of Linear Programs

Any linear program can be changed to a maximization problem as minimizing a function is the same as maximizing its negation. From now on, we'll assume that the goal of our linear programs is to maximize the objective function.

Definition 6.2. Two (maximizing) linear programs LP and LP' are said to be equivalent if for any feasible solution (x_1, \dots, x_n) to LP, there exists a feasible solution $(x'_1, \dots, x'_{n'})$ to LP' with the same objective value and for any feasible solution $(y'_1, \dots, y'_{n'})$ to LP', there exists a feasible solution (y_1, \dots, y_n) to LP with the same objective value. Thus solving LP is equivalent to LP'.

Remark. 1. LP and LP' can have a different number of decision variables i.e. we do not require $n = n'$. 2. There need not be a one-to-one correspondence between the feasible sets of LP and LP' i.e. for a feasible solution to LP there could be multiple feasible solutions with the same objective value. Similarly, in the other direction.

Remark. Even though equivalence of linear programs only requires the existence of an abstract correspondence between the feasible sets of LP and LP', in practice, one constructs linear transformations $T : \mathbb{R}^n \rightarrow \mathbb{R}^{n'}$ and $S : \mathbb{R}^{n'} \rightarrow \mathbb{R}^n$ which map the feasible set of LP to LP' and the feasible set of LP' to LP, respectively. These linear transformations need not be inverses of each other, or even isomorphisms. They only need to preserve the objective values.

Example 6.1. The following two linear programs are equivalent to each other

$$\begin{array}{ll} \text{maximize:} & x + y \\ \text{subject to:} & 0 \leq x \leq 1 \\ & 0 \leq y \leq 1 \end{array} \qquad \begin{array}{ll} \text{maximize:} & z \\ \text{subject to:} & 0 \leq z \leq 1 \end{array}$$

via the transformations $T(x, y) = x + y$ and $S(z) = (z/2, z/2)$.

Theorem 6.1. Every linear program is equivalent to a linear program in the standard form.

Proof. The proof is by providing an explicit standardization algorithm. Consider the linear program in (6.1), where we're assuming that the goal is to maximize the objective function. If it is in the standard form, then we're done. If not, then there must be a finite number of errors of the following types:

1. A linear constraint is a lower bound and has the form

$$a_{i1}x_1 + \dots + a_{in}x_n \geq b_i.$$

2. A linear constraint is an equality and has the form

$$a_{i1}x_1 + \dots + a_{in}x_n = b_i.$$

3. A variable x_j has a negativity constraint $x_j \leq 0$.

4. A variable x_j is free i.e. it is missing a sign constraint.

We fix these errors one at a time, while making sure that no new errors are introduced, thereby guaranteeing termination of the algorithm.

1. We replace the constraint with

$$-a_{i1}x_1 + \cdots - a_{in}x_n \leq -b_i.$$

2. We replace the constraint with two constraints

$$\begin{aligned} a_{i1}x_1 + \cdots + a_{in}x_n &\leq b_i \\ -a_{i1}x_1 - \cdots - a_{in}x_n &\leq -b_i, \end{aligned}$$

3. We let $y_j = -x_j$ and create a new linear program using the variables $x_1, \dots, x_{j-1}, y_j, x_{j+1}, \dots, x_n$ by replacing x_j with $-y_j$ everywhere.
4. We let $x_j = y_j - z_j$ for two new decision variables y_j and z_j with $y_j, z_j \geq 0$ and create a new linear program using the variables $x_1, \dots, x_{j-1}, y_j, z_j, x_{j+1}, \dots, x_n$ by replacing x_j with $y_j - z_j$ everywhere. We can do this because any real number can be written as a difference of two positive real numbers.

□

Exercise 6.1. Prove that the algorithm in the proof of Theorem 6.1 produces a linear program that is equivalent to the original linear program.

Part II

Duality Theory

Chapter 7

Dual Linear Program

So far we've constructed a method for solving linear programs. Our next goal is to understand the dependence of the solution on the various constants in the program. We will do this using duality theory.

We start by introducing the matrix notation. We'll let x denote the vector of decision variables, b the vector of upper bounds, c the vector of objective coefficients, and A the matrix of constraints in the standard linear program (2.1).

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}.$$

The standard linear program (2.1) can be written as follows:

$$\begin{array}{ll} \text{maximize:} & c_0 + c^T x \\ \text{subject to:} & Ax \leq b \\ & x \geq 0. \end{array}$$

Definition 7.1. The dual of this linear program is defined as the following linear program:

$$\begin{array}{ll} \text{minimize:} & c_0 + b^T y \\ \text{subject to:} & A^T y \geq c \\ & y \geq 0, \end{array} \tag{7.1}$$

where $y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$ is the vector of dual decision variables.

The original linear program is called the primal. The dual decision variables correspond to the constraints of the original linear program. By explicitly computation, we can easily prove that the following theorem.

Theorem 7.1. The dual of the (standardized) dual is equivalent to the primal.

Hence, we think of linear programs as occurring in primal-dual pairs. Every linear program has a dual and it is itself the dual of the dual. The dual decision variables correspond to the constraints of the original linear program and the primal decision variables correspond to the constraints of the dual. The meaning of the dual decision variables will become clear when we study shadow prices.

Example 7.1. The dual of (1.2) is:

$$\begin{array}{llllll} \text{minimize:} & 3.6y_1 & + & 1.5y_2 & + & y_3 \\ \text{subject to:} & 3y_1 & + & 2y_2 & + & y_3 & \geq & 4 \\ & 6y_1 & + & y_2 & + & y_3 & \geq & 3 \\ & y_1 & , & y_2 & , & y_3 & \geq & 0. \end{array} \quad (7.2)$$

The variable y_1 corresponds to “maturity”, the variable y_2 corresponds to “risk”, and the variable y_3 corresponds to “percentage”.

7.1 General Linear Program

We saw how to standardized general linear programs in Chapter 6. We need to “fix” the objective function, constraints, and signs of decision variables. To find the dual of a general linear program, we can first standardize it and then form the dual. However, doing so changes the constants. It is possible to “revert the standardization” in the resulting dual to get the same constants as the primal.

Suppose one of the linear constraint is a lower bound and has the form

$$a_{i1}x_1 + \cdots + a_{in}x_n \geq b_i.$$

To fix this, we multiply the constraint by -1 to get

$$-a_{i1}x_1 + \cdots - a_{in}x_n \leq -b_i.$$

If we then form the dual, the coefficients of the dual variable y_i will be $-a_{ij}$ instead of a_{ij} . We can replace y_i with a variable $y'_i = -y_i$ and then the resulting linear program will have the same constants as the primal but now we have $y'_i \leq 0$. Notice that this is the third type of error in a general linear program.

Suppose one of the linear constraint is a lower bound and has the form

$$a_{i1}x_1 + \cdots + a_{in}x_n = b_i.$$

To fix this, we replace the constraint with the two constraints

$$\begin{aligned} a_{i1}x_1 + \cdots + a_{in}x_n &\leq b_i, \\ -a_{i1}x_1 - \cdots - a_{in}x_n &\leq -b_i. \end{aligned}$$

If we then form the dual, we will have two dual variables y'_i and y''_i with coefficients a_{ij} and $-a_{ij}$ respectively. We can introduce a new variable, $y_i = y'_i - y''_i$ and then the resulting linear program will have the same constants as the primal but now we have y_i is free. Notice that this is the fourth type of error in a general linear program.

Because the dual of the dual is the primal, we do not need to analyze the third and fourth type of error separately. The follow table describes the modifications needed for forming the dual of a general linear program.

Primal	Dual
$a_{i1}x_1 + \cdots + a_{in}x_n \leq b_i$	$y_i \geq 0$
$a_{i1}x_1 + \cdots + a_{in}x_n \geq b_i$	$y_i \leq 0$
$a_{i1}x_1 + \cdots + a_{in}x_n = b_i$	y_i is free
$x_j \geq 0$	$a_{1j}y_1 + \cdots + a_{mj}y_m \geq c_j$
$x_j \leq 0$	$a_{1j}y_1 + \cdots + a_{mj}y_m \leq c_j$
x_j is free	$a_{1j}y_1 + \cdots + a_{mj}y_m = c_j$

Chapter 8

Weak and Strong Duality

We'll now prove several theorems relating the optimal objective values of the primal-dual pairs. Consider the standard linear program (primal) (2.1) and its

dual (7.1). We say that a vector $x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ is primal feasible if it is in the

feasible region of the primal and a vector $y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$ is dual feasible if it is in the feasible region of the dual.

8.1 Weak Duality

Theorem 8.1 (Weak duality). Suppose x is primal feasible and y is dual feasible. Then the primal objective value at x is less than or equal to the dual objective value at y .

Proof. The proof relies on analyzing the term $y^T Ax$ and follows by looking at the following sequence of inequalities:

$$\begin{aligned} b^T y &= y^T b \\ &\geq y^T (Ax) && \text{as } Ax \leq b \text{ and } y \geq 0 \\ &= (y^T Ax)^T \\ &= x^T A^T y \\ &\geq x^T c && \text{as } A^T y \geq c \text{ and } x \geq 0 \\ &= c^T x. \end{aligned}$$

□

We get several immediate corollaries from weak duality.

Corollary 8.1. If the primal is unbounded, then the dual is infeasible.

Corollary 8.2. If the dual is unbounded, then the primal is infeasible.

Corollary 8.3. If both the primal and dual have optimal solutions, then the optimal value of the primal is less than or equal to the optimal value of the dual.

We cannot say anything about the dual in the case when the primal is infeasible. Similarly, we cannot conclude anything about the existence of an optimal value of the dual in the case when the primal has an optimal solution.

8.2 Strong Duality

The tableau of the primal problem (2.1) is as follows:

$$\begin{array}{cc|c} c^T & 0 & c_0 \\ \hline A & I_m & b \end{array} \quad (8.1)$$

We can standardize the dual problem (7.1) and form its tableau:

$$\begin{array}{cc|c} -b^T & 0 & -c_0 \\ \hline -A^T & I_n & -c \end{array} \quad (8.2)$$

We will call such tableaux duals of each other. More generally, we'll say that two tableaux (of appropriate dimensions) are duals of each other if after rearranging the pivot columns, if necessary, they're of the above forms.

One can show the following theorem by explicit computation:

Lemma 8.1. Consider the two dual tableaux (8.1) and (8.2). If we pivot the first tableau about the i^{th} row and j^{th} column of A and the second tableau about the i^{th} column and j^{th} row of $-A^T$, then the resulting tableaux remain duals of each other.

Lemma 8.2. If the tableau (8.1) corresponds to an optimal solution of the primal then the tableau (8.2) corresponds to an optimal solution of the dual.

Proof. The tableau (8.1) corresponds to an optimal solution of the primal precisely when

1. (primal optimality) $c^T \leq 0$ as in this case no entering variable can be found for the primal, and
2. (primal feasibility) $b \geq 0$.

These conditions translate to

1. (dual optimality) $-b^T \leq 0$ as in this case no entering variable can be found for the dual, and

2. (dual feasibility) $-c \geq 0$.

□

Using the above two lemmas, and by explicitly running the simplex method we get the following result:

Theorem 8.2 (Strong duality). If the primal has an optimal solution then so does the dual. Moreover, they have the same optimal values.

Proof. At the optimal solution for the primal, we have a set of basic and non-basic variables. We can perform a sequence of pivot operations to get this tableau from the initial tableau. We then perform the corresponding pivots on the dual tableau. By Lemma 8.1 the resulting tableau will be dual to the primal tableau at the optimal solution. By Lemma 8.2 the dual is also optimal and has the same objective value. □

We can use the two duality theorems to come up with a fast way to verify optimality.

Theorem 8.3 (Certificate of optimality). x is an optimal solution for the primal and y is an optimal solution for the dual if and only if 1. x is primal-feasible, 2. y is dual-feasible, 3. $c^T x = b^T y$ i.e. the primal objective value at x is equal to the dual objective value at y .

Proof. (\Rightarrow)

If x and y are optimal solutions, then they are feasible by definition. By strong duality (Theorem 8.2) they must the same objective value.

(\Leftarrow)

If x and y are feasible solutions then by weak duality (Theorem 8.1) the dual objective values provide an upper bound on the primal objective value. Because this upper bound is attained at x , x must be an optimal solution of the primal. Similarly, for y . □

8.3 Complimentary slackness

There is another closely related method for verifying the correctness of solution using primal and dual slack variables.

Denote by $w = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}$ the primal slack variables and by $z = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}$ the dual slack variables. More explicitly,

$$\begin{aligned} w &= b - Ax \\ z &= -c + A^T y. \end{aligned}$$

We use this convention for z as then at a dual feasible solution $z \geq 0$.

Theorem 8.4 (Complementary slackness). Suppose x is primal feasible and y is dual feasible. Then x and y are optimal if and only if

1. for all $1 \leq j \leq n$, $x_j z_j = 0$, and
2. for all $1 \leq i \leq m$, $y_i w_i = 0$.

Proof. The proof is in two steps. We first show a weaker statement about the vanishing of two scalars and then show that the vanishing of these scalars implies complementary slackness.

Claim: x and y are optimal solutions if and only if $x^T z = 0$ and $y^T w = 0$.

We start by rewriting the slack variable:

$$\begin{array}{llll}
 & x^T z = 0 & \text{and} & y^T w = 0 \\
 \Leftrightarrow & x^T(-c + A^T y) = 0 & \text{and} & y^T(b - Ax) = 0 \\
 \Leftrightarrow & x^T c = x^T A^T y & \text{and} & y^T b = y^T Ax \\
 \Leftrightarrow & c^T x = y^T Ax & \text{and} & b^T y = y^T Ax
 \end{array}$$

Thus we are reduced to showing that x and y are optimal solutions if and only if $c^T x = y^T Ax$ and $b^T y = y^T Ax$.

(\Leftarrow) As $c^T x = y^T Ax = b^T y$, and x and y are given to be feasible, x and y are optimal by Theorem 8.3.

(\Rightarrow) By the proof of weak duality 8.1, we know that as x and y are feasible,

$$c^T x \leq y^T Ax \leq b^T y.$$

By strong duality, as x and y are optimal

$$c^T x = b^T y.$$

The only way the two can be simultaneously true is if $c^T x = y^T Ax$ and $b^T y = y^T Ax$.

Claim: $x^T z = 0$ and $y^T w = 0$ if and only if

1. for all $1 \leq j \leq n$, $x_j z_j = 0$, and
2. for all $1 \leq i \leq m$, $y_i w_i = 0$.

This follows from the fact that at a feasible solution $x, y, w, z \geq 0$. □

Remark. Complementary slackness even holds for dual of general linear programs. The definition of slack variables as above

$$\begin{aligned}
 w &= b - Ax \\
 z &= -c + A^T y.
 \end{aligned}$$

still ensures that we will have $x_j w_j \geq 0$ for all $1 \leq j \leq n$ and $y_i z_i \geq 0$ for all $1 \leq i \leq m$ and hence the above proof will go through.

If the optimal solution of the primal is at a non-degenerate BFS, then complementary slackness can be used to find the optimal solution to the dual without having to run the simplex method on it. At a non-degenerate BFS, the basic variables are non-zero. By complementary slackness, the corresponding dual variables must be 0. This gives us a system of m variables in m equations which can be solved to find the dual optimal solution.

Example 8.1. Consider the linear program (1.2) whose dual is (7.2). At the optimal solution $x_1 = 0.6$, $x_2 = 0.3$, $w_1 = 0$, $w_2 = 0$, and $w_3 = 0.1$. By complementary slackness, we must have $z_1 = 0$, $z_2 = 0$, and $y_3 = 0$ at the dual optimal solution. Plugging these in (1.2), we get a system of equations

$$\begin{aligned} 3y_1 + 2y_2 &= 4 \\ 6y_1 + y_2 &= 3 \end{aligned}$$

whose solutions are $y_1 = 2/9$ and $y_2 = 5/3$. One can check that this is indeed dual optimal by comparing the dual objective at this point to the primal objective:

$$3.6(2/9) + 1.5(5/3) + 0(1) = 3.3 = 4(0.6) + 3(0.3).$$

Exercise 8.1. We can use complementary slackness and certificate of optimality to show that the optimal solution of a standard linear program can never be attained in the interior of the feasible region.

Consider the standard linear program (2.1) and assume that $b \geq 0$ and $c \neq \vec{0}$. Let x^* be a point in the interior of the feasible region. Suppose x^* is an optimal solution to the primal. By strong duality, we know that a dual-optimal solution y^* exists. We'll show that this leads to a contradiction.

1. What can you say about the values of the decision and slack variables at x^* ?
2. What can you say about the values of the dual decision variables at y^* using complementary slackness?

We now consider two cases, depending on the sign of c , and in each case show that y^* cannot be optimal.

1. Suppose that $c_j > 0$ for some $0 \leq j \leq n$. Explain why y^* cannot be dual-feasible (and hence optimal).
2. Suppose $c \leq 0$.
 1. What can you say about the objective values at x^* and y^* ?
 2. Explain why y^* cannot be dual-optimal.

Chapter 9

Sensitivity Analysis

Linear programs are used to model real world problems. Such models are at best approximate and at worst inaccurate. As such, it is important to understand the sensitivity of our solution to changes in the model. This is broadly called sensitivity analysis. We will focus on understanding the dependence of the optimal objective value of the standard linear program (2.1) on the constants b_i and c_j .

Throughout this chapter, we'll assume that our linear programs have an optimal solution.

9.1 Dictionaries Revisited

We will start by finding a succinct way to describe the dictionary at the optimal solution. Recall that the decision and slack variables are related to each other by the Equation (3.5) which can be written as:

$$\begin{bmatrix} A & I_m \end{bmatrix} \begin{bmatrix} x \\ w \end{bmatrix} = b. \quad (9.1)$$

Let $\hat{A} := [A \quad I_m]$. We'll decompose \hat{A} using the basic and non-basic variables. Then let B be the matrix formed by combining the columns of \hat{A} corresponding to the basic variables at the optimal BFS and let N be the matrix formed by combining the columns of \hat{A} corresponding to the non-basic variables at the optimal BFS. Let x_B be the vector of basic variables and x_N be the vector of non-basic variables.

By rearranging the columns of \widehat{A} if necessary, we can rewrite (9.1) as

$$\begin{aligned} & [B \quad N] \begin{bmatrix} x_B \\ x_N \end{bmatrix} = b, \\ \Rightarrow & Bx_B + Nx_N = b, \\ \Rightarrow & Bx_B = b - Nx_N. \end{aligned}$$

When we execute the simplex method, the matrix B gets reduced to an $n \times n$ matrix with n pivots. Hence, it is invertible.

$$\Rightarrow x_B = B^{-1}b - B^{-1}Nx_N. \quad (9.2)$$

This is nothing but the dictionary at the optimal BFS.

Example 9.1. Consider Example (1.2) again. At the optimal solution w_1 and w_2 are non-basic and have the value 0, and x , y , and w_3 are basic with values 0.3, 0.6, and 0.9, respectively. Using the above notation, we have

$$\begin{aligned} B &= \begin{bmatrix} 3 & 6 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, x_B = \begin{bmatrix} x \\ y \\ w_3 \end{bmatrix}, \\ N &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, x_N = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}. \end{aligned}$$

Using Equation (9.2) the dictionary at the optimal solution becomes

$$\begin{aligned} \begin{bmatrix} x \\ y \\ w_3 \end{bmatrix} &= \begin{bmatrix} 3 & 6 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 3.6 \\ 1.5 \\ 1 \end{bmatrix} - \begin{bmatrix} 3 & 6 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ &= \begin{bmatrix} -1/9 & 2/3 & 0 \\ 2/9 & -1/3 & 0 \\ -1/9 & -1/3 & 1 \end{bmatrix} \begin{bmatrix} 3.6 \\ 1.5 \\ 1 \end{bmatrix} - \begin{bmatrix} -1/9 & 2/3 & 0 \\ 2/9 & -1/3 & 0 \\ -1/9 & -1/3 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ &= \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} - \begin{bmatrix} -1/9 & 2/3 \\ 2/9 & -1/3 \\ -1/9 & -1/3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}. \end{aligned}$$

This is precisely the dictionary (3.3) at the optimal solution.

Because we set the non-basic variables x_N to 0 at any vertex, and in particular, at the the optimal solution, using the Equation (9.2) we get the following useful result:

Lemma 9.1. Using the above notation, $B^{-1}b$ is the value of the basic variables x_B^* at the optimal solution.

9.2 Range of Optimality - Constraints

We want to analyze the change in the optimal solution as we change the constraint upper bounds b_i . It is likely that by changing b_i we change the optimal solution. However, in a good model, this change should not be abrupt. This can be achieved by requiring the set of basic and non-basic variables to remain unchanged. In this case, the equation (9.2) will still be the equation describing the dictionary at the optimal solution and the change in b_i will result in a differentiable (in fact, linear) change in x_B .

Example 9.2. Suppose we vary $b_3 = 1$ in Example (1.2). One can check that at the optimal solution w_1 and w_2 are non-basic as long as $b_3 > 0.9$. Thus we can say that our model is a good model as long as the error in b_3 is less than 0.1.

Suppose we change b_i to $b_i + \delta$, where δ is a real number, and leave all the other constants unchanged. This is equivalent to changing b to $b + \delta e_i$ where e_i is the i^{th} standard basis vector of \mathbb{R}^m . This changes equation (9.2) to

$$\begin{aligned} x_B &= B^{-1}b + \delta B^{-1}e_i - B^{-1}N x_N \\ &= B^{-1}b + \delta (B^{-1})_{\cdot i} - B^{-1}N x_N. \end{aligned}$$

where $(B^{-1})_{\cdot i}$ denotes the i^{th} column of B^{-1} .

Note that the coefficients of x_N remain unchanged. So, for this dictionary to stay optimal we only need the constants to remain non-negative i.e.

$$B^{-1}b + \delta (B^{-1})_{\cdot i} \geq 0. \quad (9.3)$$

Proposition 9.1. The range of optimality for b_i is the interval $[b_i + \delta_-, b_i + \delta_+]$ such that $B^{-1}b + \delta (B^{-1})_{\cdot i} \geq 0$ for all $\delta \in [\delta_-, \delta_+]$.

In practice, Equation (9.3) gives us m inequalities which need to be simultaneously satisfied. These give us candidate values for δ some of which are positive and some of which are negative. We then choose δ_+ to be the smallest positive value and δ_- to be the largest negative value. If δ_+ does not exist the upper bound is ∞ and if δ_- does not exist the lower bound is $-\infty$. If either δ_+ or δ_- is 0 then the linear program is degenerate. In this case, our program is very sensitive to perturbations in b_i .

Example 9.3. Let us find the range of optimality for $b_1 = 3.6$, $b_2 = 1.5$, and $b_3 = 1$ in (1.2) using our calculations in Example 9.1.

We know that

$$B^{-1} = \begin{bmatrix} -1/9 & 2/3 & 0 \\ 2/9 & -1/3 & 0 \\ -1/9 & -1/3 & 1 \end{bmatrix}.$$

Using $i = 1$ and Lemma 9.1 in Equation (9.3) we get

$$\begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} + \delta \begin{bmatrix} -1/9 \\ 2/9 \\ -1/9 \end{bmatrix} > 0$$

which gives us the inequalities

$$\begin{aligned} 0.6 + \delta(-1/9) &\geq 0 &\implies \delta &\leq 0.6(9) = 5.4 \\ 0.3 + \delta(2/9) &\geq 0 &\implies \delta &\geq -0.3(9/2) = -1.35 \\ 0.1 + \delta(-1/9) &\geq 0 &\implies \delta &\leq 0.1(9) = 0.9. \end{aligned}$$

So, $\delta_- = -1.35$ and $\delta_+ = \min(5.4, 0.9) = 0.9$ and the range of optimality for b_1 is $[3.6 - 1.35, 3.6 + 0.9] = [2.25, 4.5]$.

Using $i = 2$ and Lemma 9.1 in Equation (9.3) we get

$$\begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} + \delta \begin{bmatrix} 2/3 \\ -1/3 \\ -1/3 \end{bmatrix} > 0$$

which gives us the inequalities

$$\begin{aligned} 0.6 + \delta(2/3) &\geq 0 &\implies \delta &\geq -0.6(3/2) = -0.9 \\ 0.3 + \delta(-1/3) &\geq 0 &\implies \delta &\leq 0.3(3) = 0.9 \\ 0.1 + \delta(-1/3) &\geq 0 &\implies \delta &\leq 0.3(1) = 0.3. \end{aligned}$$

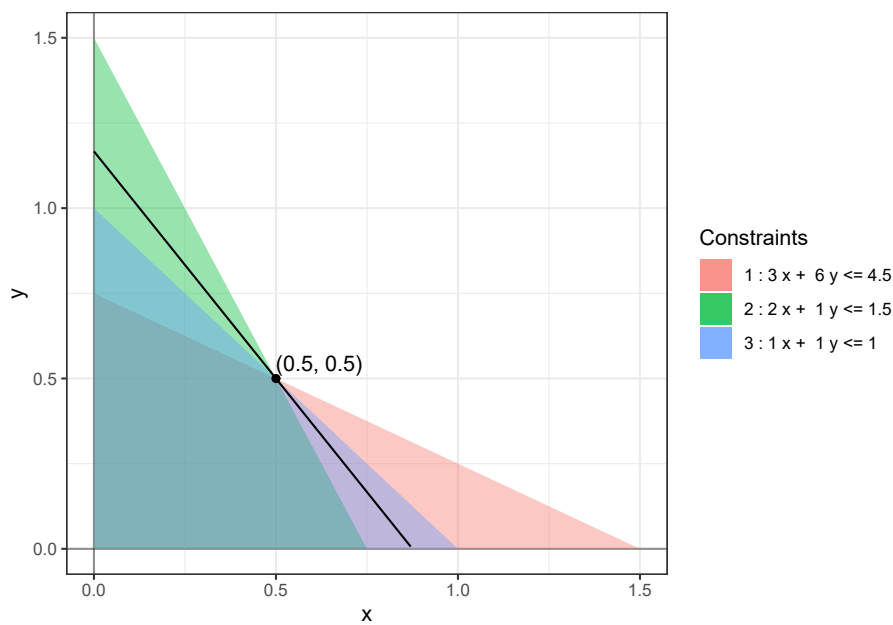
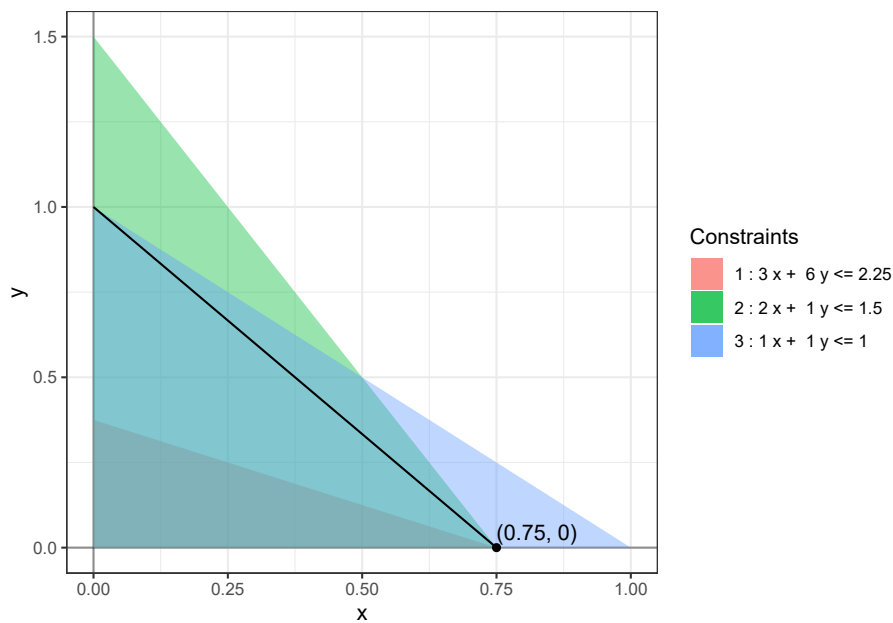
So, $\delta_- = -0.9$ and $\delta_+ = \min(0.3, 0.9) = 0.3$ and the range of optimality for b_2 is $[1.5 - 0.9, 1.5 + 0.3] = [0.6, 1.8]$.

Using $i = 3$ and Lemma 9.1 in Equation (9.3) we get

$$\begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} + \delta \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} > 0$$

which gives us $\delta \geq -0.1$ and so the range of optimality for b_3 is $[1 - 0.1, \infty) = [0.9, \infty)$.

The following figures show the optimal solutions at the extreme ends of the range of optimality of b_1 .



9.3 Shadow Prices

Assume now that neither of δ_+ or δ_- is zero. We can use Lemma 9.1 to find the rate of change of the optimal solution with respect to b_i . Call the objective

function $\mathbf{O} = c^T x$. We think of \mathbf{O} as being a function of b_i , c_j , and a_{ij} . Using Lemma 9.1 we get

$$\begin{aligned}\frac{\partial x_{B_j}}{\partial b_i} &= j^{th} \text{ row of } \frac{\partial B^{-1}b}{\partial b_i} \\ &= (B^{-1})_{ji}\end{aligned}$$

where x_{B_j} denotes the j^{th} basic variable at the optimal solution. And

$$\frac{\partial x_N}{\partial b_i} = 0$$

as the non-basic variables remain 0 when we perturb b_i within the range of optimality. Using these we can find the rate of change of the optimal solution \mathbf{O} with respect to b_i . We first start by re-indexing the variables and objective coefficients using the basic and non-basic variables.

$$\begin{aligned}\mathbf{O} &= c^T x \\ &= c_B^T x_B + c_N^T x_N \\ \Rightarrow \frac{\partial \mathbf{O}}{\partial b_i} &= c_B^T \frac{\partial x_B}{\partial b_i} + c_N^T \frac{\partial x_N}{\partial b_i} \\ &= c_B^T (B^{-1})_{-i}\end{aligned}$$

By strong duality, we know that the primal objective value equals the dual objective value i.e.

$$\mathbf{O} = b_1 y_1 + \cdots + b_m y_m$$

So, $\partial \mathbf{O} / \partial b_i = y_i$. Because of this result, y_i is also called the shadow price or the marginal cost of the i^{th} constraint. Using the rate of change calculation above, we get

$$y_i = c_B^T (B^{-1})_{-i}.$$

We can combine all the above coordinates into a single vector to get the following result.

Theorem 9.1. For a non-degenerate linear program, the dual optimal solution is given by $(B^{-1})^T c_B$.

This theorem provides yet another method of finding the dual optimal solution without having to solve the dual linear program.

Example 9.4. For the linear program (1.2), the objective function is

$$\begin{aligned}\mathbf{O} &= 4x + 3y \\ &= 4x + 3y + 0w_3 + 0w_1 + 0w_2\end{aligned}$$

So, $c_B = \begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix}$ and $c_N = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. Using the value of B^{-1} calculated above, we get

$$\begin{aligned} y &= (B^{-1})^T c_B \\ &= \begin{bmatrix} -1/9 & 2/9 & -1/9 \\ 2/3 & -1/3 & -1/3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 2/9 \\ 5/3 \\ 0 \end{bmatrix}. \end{aligned}$$

To check that this is indeed dual-optimal, we calculate the dual-objective value at this solution

$$\begin{aligned} b^T y &= 3.6(2/9) + 1.5(5/3) + 0(1) \\ &= 3.3 \end{aligned}$$

which equals the optimal objective value of the primal. One can check that this solution is also dual-feasible and hence is the dual-optimal solution by Certificate of Optimality (Theorem 8.3).

9.4 Sensitivity analysis - Objective

We can ask the same questions about the change in the objective coefficients - how far can we change the objective coefficient c_j without changing the set of basic and non-basic variables at the optimal solution. Note that in this case we are not changing the constraints and therefore the feasible region. So, this is the same as asking - how far can we change the objective coefficient c_j without changing the optimal solution.

We can redo the entire analysis for the objective coefficients from scratch. However, we can also notice that performing sensitivity analysis on the objective coefficients of the primal is the same as performing sensitivity analysis on the constraints of the dual. Consider the standardized dual

$$\begin{aligned} \text{maximize:} & \quad -c_0 - b^T y \\ \text{subject to:} & \quad -A^T y \leq -c \\ & \quad y \geq 0, \end{aligned}$$

Using Equation (9.3) for this dictionary we get the range of optimality for $-c_j$ to be

$$(-B_d)^{-1}(-c) + \delta(-B_d^{-1})_{\cdot j} \geq 0. \quad (9.4)$$

where $-B_d$ is formed by combining the dual-basic columns of $[-A^T \ I_n]$ and so B_d is formed by combining the dual-basic columns of $[A^T \ -I_n]$. This simplifies to

$$B_d^{-1}c - \delta(B_d^{-1})_{\cdot j} \geq 0. \quad (9.5)$$

Note that this is the range of optimality for $-c_j$. To get the range of optimality for c_j we need to replace δ by $-\delta$ to get

$$B_d^{-1}c + \delta(B_d^{-1})_{\cdot j} \geq 0. \quad (9.6)$$

Finally, by Lemma 9.1, $B_d^{-1}c$ is the vector of values of the dual basic variables. Hence we get,

Proposition 9.2. The range of optimality for c_j is the interval $[c_j + \delta_-, c_j + \delta_+]$ such that $y_B^* + \delta(B_d^{-1})_{\cdot j} \geq 0$ for all $\delta \in [\delta_-, \delta_+]$.

By a happy accident all the negative signs have cancelled out and the equation for finding the range of optimality for the objective coefficients is exactly the same as the one for finding the range of optimality for the constraints.

Theorem 9.2. The range of optimality for the constraints and the objective functions can be computed using the following formulae:

Range of optimality for b_i :

$$x_B^* + \delta(B^{-1})_{\cdot i} \geq 0. \quad (9.7)$$

where B is formed by combining the primal-basic columns of $[A \quad I_m]$, and

Range of optimality for c_j :

$$y_B^* + \delta(B_d^{-1})_{\cdot j} \geq 0. \quad (9.8)$$

where B_d is formed by combining the dual-basic columns of $[A^T \quad -I_n]$.

Finally, to find B_d we note that by Complementary Slackness (Theorem 8.4), if the linear program is non-degenerate then the dual basic variables correspond to the primal non-basic variables (as the dual basic variables must be non-zero). This statement is true even for degenerate linear programs but in this case the proof is more subtle and requires the use of Strong duality (Theorem 8.2).

Example 9.5. Consider the linear program (1.2) again. At the optimal solution, as w_1 and w_2 are non-basic, the corresponding dual variables (y_1 and y_2) will be basic for the dual linear program (7.2). This gives us

$$\begin{aligned} B_d &= \begin{bmatrix} 3 & 2 \\ 6 & 1 \end{bmatrix} \\ \Rightarrow B_d^{-1} &= \begin{bmatrix} -1/9 & 2/9 \\ 2/3 & -1/3 \end{bmatrix}. \end{aligned}$$

From Example 9.4 we know that $y_1 = 2/9$ and $y_2 = 5/3$ at the dual optimal solution. Using these, we can now find the range of optimality for the objective coefficients.

To find the range of optimality for $c_1 = 4$ we solve

$$\begin{bmatrix} 2/9 \\ 5/3 \end{bmatrix} + \delta \begin{bmatrix} -1/9 \\ 2/3 \end{bmatrix} \geq 0$$

which gives us the inequalities

$$\begin{aligned} 2/9 + \delta(-1/9) &\geq 0 &\implies \delta &\leq 2 \\ 5/3 + \delta(2/3) &\geq 0 &\implies \delta &\geq -5/2 \end{aligned}$$

So, $\delta_- = -5/2$ and $\delta_+ = 2$ and the range of optimality for c_1 is $[4 - 5/2, 4 + 2] = [1.5, 6]$.

To find the range of optimality for $c_2 = 3$ we solve

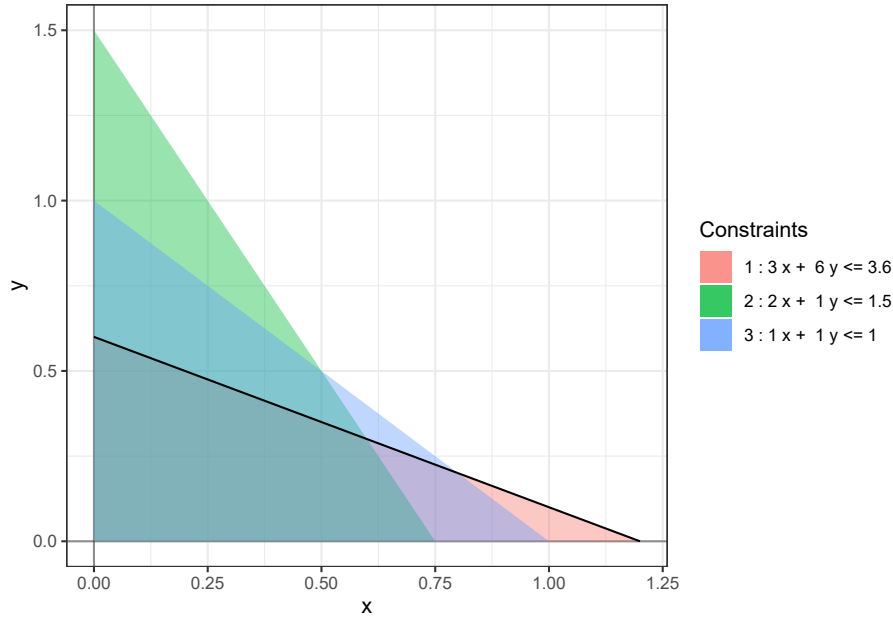
$$\begin{bmatrix} 2/9 \\ 5/3 \end{bmatrix} + \delta \begin{bmatrix} 2/9 \\ -1/3 \end{bmatrix} \geq 0$$

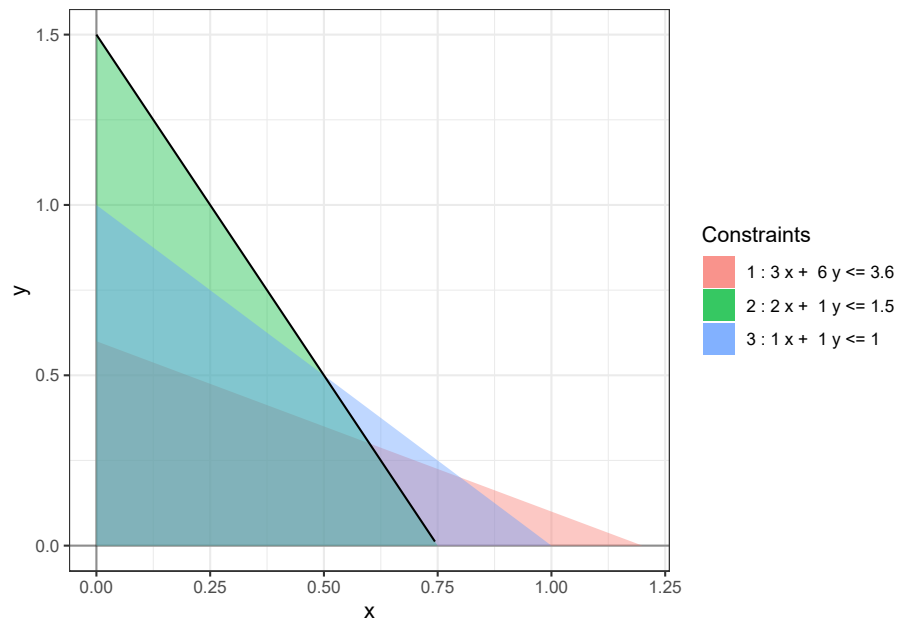
which gives us the inequalities

$$\begin{aligned} 2/9 + \delta(2/9) &\geq 0 &\implies \delta &\geq -1 \\ 5/3 + \delta(-1/3) &\geq 0 &\implies \delta &\leq 5 \end{aligned}$$

So, $\delta_- = -1$ and $\delta_+ = 5$ and the range of optimality for c_2 is $[3 - 1, 3 + 5] = [2, 8]$.

The following figures show the (infinitely many) optimal solutions at the extreme ends of the range of optimality of c_1 .





Part III

Non-Linear Programming

Chapter 10

Convex Programming

We will now move toward generalizing the concepts from linear programming to non-linear optimization questions. We'll only touch several subjects briefly without going into too much depth. We'll start with convex optimization, which is a direct generalization of linear programming.

Definition 10.1. We say that a subset S of \mathbb{R}^n is called convex if for any points x_1 and x_2 in S , the line segment connecting them also lies in S . More precisely, we say that S is convex if for all x_1 in S , x_2 in S , and $t \in [0, 1]$, the point $(1-t)x_1 + tx_2$ also lies in S .

Exercise 10.1. Show that the feasible region of a linear program is convex.

Definition 10.2. We say that function $f : S \rightarrow \mathbb{R}$ is convex if for all x_1 in S , x_2 in S , and $t \in [0, 1]$, the inequality $f((1-t)x_1 + tx_2) \leq (1-t)f(x_1) + tf(x_2)$ holds. Geometrically, this is saying that any line segment connecting two points on the graph of f lies above the graph.

The following theorem provides a quick way to check if a function is convex and to come up with examples of convex functions.

Theorem 10.1. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a C^2 function. f is convex if and only if the Hessian matrix of f is positive semi-definite. In particular, when $n = 1$, f is convex if and only if $f'' \geq 0$.

Definition 10.3. An optimization problem

$$\begin{array}{ll} \text{maximize:} & f(x) \\ \text{subject to:} & x \in S \end{array} \tag{10.1}$$

is called convex if S is a closed and convex subset of \mathbb{R}^n and $f : S \rightarrow \mathbb{R}$ is a convex function.

Exercise 10.2. Show that a linear program is a convex optimization problem.

The following theorem is an example of how theorems about linear programs generalize to convex programs.

Theorem 10.2. The convex optimization problem (10.1) either has an optimal solution on the boundary of S or is unbounded.

Proof. Let x be a point in the interior of S . It suffices to show that either there is some point on the boundary of S with an objective value that is greater than or equal to the objective value of x or the problem is unbounded.

Let x_1 be any point on the boundary of S . Draw a ray starting at x_1 in the direction of x . As S is closed and convex, two cases are possible:

1. The ray intersects the boundary of S in exactly one more point, say x_2 .
2. The ray does not intersect the boundary of S and the entire ray is contained within S .

Case 1:

We will show that either $f(x) \leq f(x_1)$ or $f(x) \leq f(x_2)$. We prove this by contradiction. Suppose $f(x) > f(x_1)$ and $f(x) > f(x_2)$. We know that $x = (1-t)x_1 + tx_2$ for some $t \in (0, 1)$. Multiplying the first inequality by $(1-t)$ and the second by t and adding them together, we get

$$\begin{aligned} (1-t)f(x) + tf(x) &> (1-t)f(x_1) + tf(x_2) \\ \implies f(x) &> (1-t)f(x_1) + tf(x_2), \end{aligned}$$

which contradicts the convexity of f .

Case 2:

If $f(x_1) \geq f(x)$, we're done. Suppose this is not the case. We'll show that the problem is unbounded.

For $0 < t \leq 1$, let y_t denote the point on the ray such that

$$x = (1-t)x_1 + ty_t.$$

For example, $y_1 = x$ and $y_{1/2}$ is the point on the ray for which x is midpoint of x_1 and $y_{1/2}$. As t decreases, y_t will move farther and farther away from x . Note that all of y_t lie inside S . By convexity of f , we know that

$$\begin{aligned} f(x) &\leq (1-t)f(x_1) + tf(y_t) \\ \implies f(x) - (1-t)f(x_1) &\leq tf(y_t) \\ \implies t^{-1}f(x) - (t^{-1} - 1)f(x_1) &\leq f(y_t) \\ \implies t^{-1}(f(x) - f(x_1)) + f(x_1) &\leq f(y_t) \end{aligned}$$

As $f(x) > f(x_1)$, the left hand side tends to ∞ as t decreases and the optimization problem is unbounded. \square

The following corollary follows by applying Extreme Value Theorem to the above result.

Theorem 10.3. If S is bounded (in addition to being closed and convex), then the convex optimization problem (10.1) has an optimal solution on the boundary of S .

Chapter 11

Separation Theorems

We'll next prove a couple of geometric results that are equivalent to strong duality theorem (Theorem 8.2).

11.1 Farkas' Lemma

Theorem 11.1 (Farkas' lemma). Let A be an $m \times n$ matrix and b be a vector in \mathbb{R}^m . Exactly one of the following systems has a solution:

$$Ax = b \tag{11.1}$$

$$x \geq 0. \tag{11.2}$$

$$y^T A \geq 0 \tag{11.3}$$

$$y^T b < 0. \tag{11.4}$$

Proof. We'll prove Farkas' lemma using strong duality. Consider the following linear program:

$$\begin{array}{ll} \text{maximize:} & 0 \\ \text{subject to:} & Ax = b \\ & x \geq 0. \end{array} \tag{11.5}$$

The optimal solution to the linear program (11.5) is a feasible solution to (11.2). The dual to this linear program is

$$\begin{array}{ll} \text{minimize:} & y^T b \\ \text{subject to:} & y^T A \geq 0. \end{array} \tag{11.6}$$

Case 1: Suppose (11.2) has a solution.

In this case, (11.5) has an optimal solution. By strong duality, (11.6) also has an optimal solution with optimal objective 0. So, the minimum value of $y^T b$ is 0 and hence the system (11.4) does not have a solution.

Case 2: Suppose (11.2) does not have a solution.

In this case, (11.5) has no optimal solution. By strong duality, neither does (11.6). So, (11.6) is either infeasible or unbounded. As $y = 0$ is a feasible solution to (11.6) it cannot be infeasible, and hence it must be unbounded. But this means that the value of $y^T b$ can be made arbitrarily small, and in particular, can be made negative. Hence, the system (11.4) has a solution. \square

We can interpret Farkas' lemma geometrically using convex cones and separating hyperplanes.

11.2 Separating Hyperplane Theorem

Definition 11.1. The convex cone of a finite set of vectors in \mathbb{R}^m is the set of positive linear combinations of vectors in the set.¹

$$C_+(v_1, \dots, v_n) := \{c_1 v_1 + \dots + c_n v_n \mid c_i \geq 0\}.$$

Definition 11.2. A hyperplane in \mathbb{R}^m is the set of solutions to a single linear equation. The complement of a hyperplane in \mathbb{R}^n consists of two connected components. The closures of these components are called half-spaces.

If the equation of the hyperplane is given by $b^T y = b_0$, where b is a vector in \mathbb{R}^m and $b_0 \in \mathbb{R}$, then the corresponding two half-spaces are described by $b^T y \leq b_0$ and $b^T y \geq b_0$.

Definition 11.3. We say that a hyperplane H separates two subsets S_1 and S_2 of \mathbb{R}^m if S_1 and S_2 do not intersect H and belong to the two different half-spaces of H .

Using convex cones and separating hyperplanes, we can reinterpret 11.1 as follows.

Theorem 11.2 (Geometric version of Farkas' lemma). Let v_1, \dots, v_n, b be vectors in \mathbb{R}^m . Exactly one of the following statements is true:

1. Either b lies inside $C_+(v_1, \dots, v_n)$, or
2. There is a hyperplane H that separates b from $C_+(v_1, \dots, v_n)$.

A point b and a convex cone $C_+(v_1, \dots, v_n)$ are convex subsets of \mathbb{R}^m . The statement of Farkas' lemma 11.2 can be generalized to arbitrary convex sets using metric topology. For now, we'll generalize it to convex polyhedra.

¹One can also define convex cones for infinite sets of vectors. In this case, we need to take the closure of the set of positive linear combinations.

Definition 11.4. Intersection of finitely many half-spaces is called a convex polyhedron.

So, a convex polyhedron is the set of solutions to a system of linear inequalities $Ax \leq b$. But this set is precisely the feasible region of a linear program! The class of convex polyhedra is very large and all geometric “linear” convex objects, like points, lines, planes, half-spaces, hyperplanes, convex cones, etc. can be realized as convex polyhedra.

The following is an extension of Farkas’ lemma to convex polyhedra (proof in the exercises below).

Theorem 11.3 (Separating Hyperplane Theorem). Any two non-empty, disjoint, convex polyhedra in \mathbb{R}^m can be separated by a hyperplane.

Exercise 11.1. Prove the following variant of Farkas’ lemma: Let A be an $m \times n$ matrix and let b be a vector in \mathbb{R}^m . Exactly one of the following systems has a solution:

1. $Ax \leq b$,
2. $y^T b < 0$, $y^T A = 0$, and $y \geq 0$.

Exercise 11.2. Let P_1 be the convex polyhedron defined by $A_1 x \leq b_1$ and let P_2 be the convex polyhedron defined by $A_2 x \leq b_2$ where A_1 , A_2 , b_1 , and b_2 have sizes $m_1 \times n$, $m_2 \times n$, m_1 , and m_2 , respectively. Suppose the system

$$\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} x \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad (11.7)$$

does not have a solution.

1. Apply the above variant of Farkas’ lemma to (11.7) to obtain vectors y_1 , y_2 of size m_1 and m_2 , respectively. Define $c^T := y_1^T A_1$, $d_1 := y_1^T b_1$, and $d_2 := -y_2^T b_2$.
2. Show that $d_1 < d_2$.
3. Show that $A_1 x \leq b_1$ implies $c^T x \leq d_1$.
4. Show that $A_2 x \leq b_2$ implies $c^T x \geq d_2$.
5. Conclude that there exists a hyperplane separating P_1 and P_2 .

11.3 Equivalence with Strong Duality

It is possible to prove Strong Duality using Farkas’ lemma, which itself can be proven using metric topology. So, Strong Duality, Farkas’ lemma, and Separating Hyperplane Theorem should all be thought of as equivalent to each other. We provide below a proof of Strong Duality (Theorem 8.2) using Farkas’ lemma.

Proof. Consider the system of equations

$$\begin{aligned}
Ax &\leq b \\
-A^T y &\leq -c \\
x, y &\geq 0 \\
-c^T x + b^T y &\leq 0.
\end{aligned} \tag{11.8}$$

Suppose this system has a feasible solution. The first three equations are equivalent to saying that x is primal-feasible and y is dual-feasible. If such a solution exists, by Weak Duality (Theorem 8.1) we know that $c^T x \leq b^T y$. So, the only way the fourth inequality is satisfied is if $c^T x = b^T y$ i.e. the primal-objective value at x equals the dual-objective value at y . But by Weak Duality, these must then be the optimal solutions thereby proving Strong Duality. So, it suffices to show that (11.8) has a feasible solution if the primal has an optimal solution.

We prove this by contradiction. Suppose the primal has an optimal solution but (11.8) does not have a feasible solution. We rewrite the system as

$$\begin{aligned}
\begin{bmatrix} A & 0 \\ 0 & -A^T \\ -c^T & b^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &\leq \begin{bmatrix} b \\ -c \\ 0 \end{bmatrix} \\
x, y &\geq 0.
\end{aligned}$$

By (a variant of) Farkas' lemma, the following dual system must have a solution.

$$\begin{aligned}
[z^T \quad w^T \quad t] \begin{bmatrix} b \\ -c \\ 0 \end{bmatrix} &< 0 \\
[z^T \quad w^T \quad t] \begin{bmatrix} A & 0 \\ 0 & -A^T \\ -c^T & b^T \end{bmatrix} &\geq 0 \\
z, w, t &\geq 0.
\end{aligned}$$

which can be rewritten as

$$\begin{aligned}
z^T b &< w^T c \\
z^T A &\geq t c^T \\
t b^T &\geq w^T A^T \\
z, w, t &\geq 0.
\end{aligned}$$

By combining the second and third equations, we get

$$t z^T b \geq z^T A w \geq t c^T w.$$

If $t > 0$, this contradicts the first equation and we're done. So suppose $t = 0$. Plugging in $t = 0$, we get

$$\begin{aligned} z^T b &< w^T c \\ z^T A &\geq 0 \\ 0 &\geq w^T A^T \\ z, w &\geq 0. \end{aligned}$$

which can be rewritten as the matrix equation

$$\begin{aligned} [z^T \quad w^T] \begin{bmatrix} b \\ -c \end{bmatrix} &< 0 \\ [z^T \quad w^T] \begin{bmatrix} A & 0 \\ 0 & -A^T \end{bmatrix} &\geq 0 \\ z, w &\geq 0. \end{aligned}$$

Because this system has a solution, by applying (variant of) Farkas' lemma again, the following system cannot have a solution.

$$\begin{aligned} \begin{bmatrix} A & 0 \\ 0 & -A^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &\leq \begin{bmatrix} b \\ -c \end{bmatrix} \\ x, y &\geq 0. \end{aligned}$$

We had assumed that the primal is feasible. The only way the above system is infeasible is if there is no solution to the system

$$\begin{aligned} A^T y &\geq c \\ y &\geq 0. \end{aligned}$$

Applying (variant of) Farkas' lemma yet again we see that the dual system

$$\begin{aligned} z^T c &< 0 \\ z^T A^T &\geq 0 \\ z &\leq 0 \end{aligned}$$

must have a solution. But now, if x is any primal-feasible solution then so is $x - \alpha z$ for any positive constant α and the objective value of the primal at $x - \alpha z$ can be made arbitrarily large by increasing α thereby contradicting the fact that primal has an optimal solution (and hence is bounded). \square

Chapter 12

Interior Point Methods

Throughout this chapter we will let f denote a twice differentiable function from \mathbb{R}^n to \mathbb{R} .

12.1 Gradient Descent

We will start by trying to solve the following unconstrained optimization problem:

$$\text{minimize: } f(x)$$

where x is any vector in \mathbb{R}^n . Gradient descent is a simple algorithm for solving this problem using basic differential calculus. It relies on the fact that the negative of the gradient points in the direction in which the function decreases the fastest. So the general principle is to move in the direction of the negative gradient until the function is no longer decreasing. More precisely, we create a sequence of guesses using the following GD recurrence relation:

$$x_{k+1} = x_k - t_k \nabla f(x_k)$$

where t_k is the “step size” for the k^{th} iteration and can be chosen to be a small constant or some function of k , x_k , $f(x_k)$, or $\nabla f(x_k)$.

There are several issues with this technique:

1. If the function has multiple local minima, then the sequence might converge to a non-absolute minima depending on the starting guess and the choice of step sizes.
2. If the step sizes are chosen to be too large, then the sequence can completely miss the minima and may not converge.
3. If the step sizes are chosen to be too small, then the sequence may take a long time to converge.

Unfortunately, there is no easy way to resolve these issues. In practice, either we need some additional information about the function and its gradient or we proceed by trial and error to find step sizes that work. Even then there is no guarantee that the algorithm will converge to an absolute minima and not a local minima, if at all. In spite of these issues, because of its simplicity, ease of implementation, and good convergence properties in practice, Gradient Descent is a very popular algorithm for solving unconstrained optimization problems.

12.2 Interior Point Method

Gradient descent can be modified to solve constrained optimization problems by introducing barrier functions. Consider the following problem:

$$\begin{aligned} &\text{minimize:} && f(x) \\ &\text{subject to:} && g_i(x) \geq 0, \quad \text{for } 1 \leq i \leq m. \end{aligned} \tag{12.1}$$

We can apply GD to this problem and find a critical point for $f(x)$. However, this might not answer the optimization question for two reasons:

1. The critical point might not be in the feasible region.
2. The optimal solution might not be obtained at a critical of $f(x)$ and could lie on the boundary $g(x) = 0$ of the feasible region.

Gradient descent algorithm only sees the objective function and does not know about the constraints. So, we modify the objective function to include the constraints using barrier functions.

Definition 12.1. A barrier function is a differentiable function b from $(0, \infty)$ to \mathbb{R} that has the property $\lim_{x \rightarrow 0^+} b(x) = \infty$.

We'll use the barrier function $-\ln x$. Using a small positive parameter μ , we create a new objective function:

$$f_\mu(x) := f(x) - \mu \sum_{i=1}^m \ln(g_i(x)).$$

Consider the unconstrained optimization problem

$$\text{minimize: } f_\mu(x).$$

Because the domain of $\ln x$ is $(0, \infty)$, any critical point of $f_\mu(x)$ must lie in the feasible region of (12.1). Let x_μ^* be a solution of the above problem. For sufficiently good functions f and g_i , x_μ^* exists and is continuous in μ and $\lim_{\mu \rightarrow 0^+} x_\mu^* = x^*$ where x^* is an optimal solution to (12.1). This assumption is valid, for example, for linear programs. With this assumption, we now have a method for solving (12.1):

- Start with a small $\mu_0 > 0$ and optimize $f_{\mu_0}(x)$ using GD. Call the solution $x_{\mu_0}^*$.

- Repeat the following until the sequence $x_{\mu_k}^*$ stabilizes sufficiently:
 1. Set $\mu_{k+1} = \mu_k - \delta_k$ for some small δ_k .
 2. Using $x_{\mu_k}^*$ as a starting point for GD, optimize $f_{\mu_{k+1}}(x)$. Call the solution $x_{\mu_{k+1}}^*$.
 3. Increase k to $k + 1$.

This method is a very simplified interior point method. The sequence of points x_{μ} is called the central path.

Example 12.1. We can compute the central path explicitly for some simple examples. Consider the optimization problem:

$$\begin{aligned} &\text{minimize: } (x+1)^2 \\ &\text{subject to: } x \geq 0. \end{aligned} \tag{12.2}$$

We can calculate the central path as follows:

$$f_{\mu}(x) := (x+1)^2 - \mu \ln(x).$$

The critical points for this function can be obtained by setting the derivative to 0.

$$\begin{aligned} f'_{\mu}(x) &= 0 \\ \implies 2(x+1) - \frac{\mu}{x} &= 0 \\ \implies x^2 + x - \mu/2 &= 0 \\ \implies x &= \frac{-1 \pm \sqrt{1+2\mu}}{2}. \end{aligned}$$

Only one of the two satisfy $x \geq 0$, so we get

$$x_{\mu}^* = \frac{-1 + \sqrt{1+2\mu}}{2}.$$

This is the central path. Taking the limit as $\mu \rightarrow 0$ we get,

$$\begin{aligned} \lim_{\mu \rightarrow 0^+} x_{\mu}^* &= \lim_{\mu \rightarrow 0^+} \frac{-1 + \sqrt{1+2\mu}}{2} \\ &= 0, \end{aligned}$$

which is indeed the optimal solution for the optimization problem (12.2).

Chapter 13

KKT conditions

We can use the central path to derive some conditions for optimality. Consider the optimization problem (12.1) again. As before, we'll assume that there is a central path x_μ^* converging to an optimal solution x^* . Because x_μ^* is a critical point of $f_\mu(x)$, we get

$$\begin{aligned}\nabla_x f_\mu(x_\mu^*) &= 0 \\ \Rightarrow \nabla f(x_\mu^*) - \sum_{i=1}^m \frac{\mu}{g_i(x_\mu^*)} \nabla g_i(x_\mu^*) &= 0\end{aligned}$$

Applying $\lim_{\mu \rightarrow 0^+}$ to both sides, we get

$$\lim_{\mu \rightarrow 0^+} \nabla f(x_\mu^*) - \sum_{i=1}^m \lim_{\mu \rightarrow 0^+} \frac{\mu}{g_i(x_\mu^*)} \nabla g_i(x_\mu^*) = 0$$

which simplifies to

$$\nabla f(x^*) = \sum_{i=1}^m \lambda_i \nabla g_i(x^*)$$

where $\lambda_i = \lim_{\mu \rightarrow 0^+} \frac{\mu}{g_i(x_\mu^*)}$ for $1 \leq i \leq m$. (There are several assumptions here about the existence and convergence of limits that we're brushing under the rug.) We can further analyze the constants λ_i .

$$\begin{aligned}\frac{\mu}{g_i(x_\mu^*)} \cdot g_i(x_\mu^*) &= \mu \\ \Rightarrow \lim_{\mu \rightarrow 0^+} \frac{\mu}{g_i(x_\mu^*)} \cdot \lim_{\mu \rightarrow 0^+} g_i(x_\mu^*) &= \lim_{\mu \rightarrow 0^+} \mu \\ \Rightarrow \lambda_i g_i(x^*) &= 0.\end{aligned}$$

Finally, because $g_i(x_\mu^*) \geq 0$ and $\mu > 0$, we must have

$$\lambda_i \geq 0.$$

To summarize, if x^* is an optimal solution to (12.1) and there exists a central path x_μ^* converging to x^* , then the following conditions are satisfied:

$$\begin{aligned}\nabla f(x^*) &= \sum_{i=1}^m \lambda_i \nabla g_i(x^*) \\ \lambda_i g_i(x^*) &= 0 \\ \lambda_i &\geq 0.\end{aligned}$$

These are called the KKT conditions. We can combine the above derivation with Lagrange multipliers to get the following generalization:

Theorem 13.1 (KKT conditions). Consider the optimization problem:

$$\begin{aligned}\text{minimize:} & \quad f(x) \\ \text{subject to:} & \quad g_i(x) \geq 0, \quad \text{for } 1 \leq i \leq m \\ & \quad h_i(x) = 0, \quad \text{for } 1 \leq i \leq k.\end{aligned}$$

Assume that this problem has an optimal solution x^* and there is a central path x_μ^* converging to x^* . Then under certain regularity conditions, the following conditions are satisfied:

$$\begin{aligned}\nabla f(x^*) &= \sum_{i=1}^m \lambda_i \nabla g_i(x^*) + \sum_{i=1}^k \lambda'_i \nabla h_i(x^*) \\ \lambda_i g_i(x^*) &= 0, \quad \text{for } 1 \leq i \leq m \\ \lambda_i &\geq 0, \quad \text{for } 1 \leq i \leq m.\end{aligned}$$

The conditions for necessity, sufficiency and the regularity conditions are quite non-trivial and are beyond the scope of this course.

Example 13.1. Consider Example 12.1 again. The KKT conditions for this example are

$$\begin{aligned}2(x+1) &= \lambda \\ \lambda x &= 0 \\ \lambda &\geq 0.\end{aligned}$$

These need to be satisfied in addition to the original constraint $x \geq 0$. The only solution to this system is $x = 0$, which is indeed the optimal solution.

Example 13.2. Consider the linear program

$$\begin{aligned}\text{minimize:} & \quad c^T x \\ \text{subject to:} & \quad Ax \geq b\end{aligned}$$

The KKT conditions for this problem are

$$\begin{aligned} c &= A^T \lambda \\ x_i \lambda_i &= 0, \text{ for } 1 \leq i \leq n, \\ \lambda &\geq 0. \end{aligned}$$

This is precisely the dual of the linear programming problem and complementary slackness conditions. Thus the KKT conditions recover duality in the linear programming sense.

Exercise 13.1. Find the KKT conditions for the standard linear program

$$\begin{aligned} \text{maximize: } & c^T x \\ \text{subject to: } & Ax \leq b \\ & x \geq 0. \end{aligned}$$

Explain how these conditions recover the dual constraints and complementary slackness.

Part IV

Applications

Chapter 14

L1-Regression

Linear regression is a method for modelling the relationship between outputs and inputs using linear functions. Suppose that the data set consists of the points (x_i, y_i) with $i = 1, 2, \dots, N$ where each x_i is itself a vector in \mathbb{R}^m . We want to find a function $f : \mathbb{R}^m \rightarrow \mathbb{R}$ such that $f(x_i) \approx y_i$. In linear regression, we suppose that the function f is of the form

$$f(x) = \beta^T x + \beta_0,$$

where β is a vector in \mathbb{R}^m and β_0 is a constant. Our goal is to estimate β and β_0 . There is of course, no reason that there is a linear relation between the inputs x_i and the outputs y_i . So our is simply to find the best estimate. We can define the error in estimation to be the vector ε whose components are

$$\varepsilon_i = |y_i - f(x_i)|.$$

We wish to “minimize” the vector in ε . There are multiple measures one can define on the space of these errors and minimizing each provides us a best estimate in the sense of that particular measure.

The most commonly used measure is the Euclidean distance or the L^2 -norm. We define

$$\|\varepsilon_i\|_2 = \sum_{i=1}^N (y_i - f(x_i))^2.$$

In L^2 -regression we try to find the function f that minimizes this quantity. There are several reasons why this measure is most commonly used, one of which being that there is a closed form for the solution using the normal equation. However, the solution obtained from L^2 -regression is very susceptible to outliers. By comparison, the L^1 -measure, defined below, is more robust to outliers.

$$\|\varepsilon_i\|_1 = \sum_{i=1}^N |y_i - f(x_i)|.$$

The biggest downside of L^1 is that it does not have a closed form solution. However, it is possible to reduce the problem of finding the best L^1 -estimate to linear programming.

Our goal is to solve the following unconstrained minimization problem

$$\text{minimize: } \sum_{i=1}^N |y_i - (\beta^T x_i + \beta_0)|,$$

where our variables are β and β_0 . As stated, this problem is not a linear program. However, it is equivalent to the following linear program:

$$\begin{aligned} \text{minimize: } & \sum_{i=1}^N \varepsilon_i \\ \text{subject to: } & y_i - (\beta^T x_i + \beta_0) < \varepsilon_i \\ & y_i - (\beta^T x_i + \beta_0) > -\varepsilon_i. \end{aligned}$$

Here the decision variables are β (which is a vector in \mathbb{R}^m), β_0 , and ε_i for $1 \leq i \leq N$.

Chapter 15

Network Flow

One very important application of linear programming is to finding max-flows in graphs. The setup is as follows: we have a simple directed graph $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ is the set of directed edges. Each edge has a non-negative capacity given by a function $c : E \rightarrow \mathbb{R}_{\geq 0}$. For simplicity, we can assume that capacity is a function $c : V \times V \rightarrow \mathbb{R}_{\geq 0}$ such that $c(v, v') = 0$ if there is no edge going from v to v' . There are two special vertices called source and sink denoted s and t , respectively. The source s has the property that every edge connected to the source points away from it i.e. there are no incoming edges and the sink t has the property that every edge connected to the sink points toward it i.e. there are no outgoing edges. Such a graph is called a network.

A flow is a function $f : E \rightarrow \mathbb{R}$ with the property that $f \leq c$. As before, we can assume that f is a function $V \times V \rightarrow \mathbb{R}$. A flow is said to be balanced if at each vertex the incoming flow equals the outgoing flow. The max-flow question is to find a balanced flow that maximizes the net outflow from the source. More precisely,

$$\begin{array}{ll} \text{maximize:} & \sum_{v \in V} f(s, v) \\ \text{subject to:} & f(v, w) \leq c(v, w) \quad \text{for all } v, w \in V \\ & \sum_{w \in V} f(v, w) = \sum_{w \in V} f(w, v) \quad \text{for all } v \in V. \end{array}$$

This is very clearly a linear program that can be solved using the simplex method. Note however, that several faster algorithms exist for solving this problem.

15.1 Min-cut

The dual of the max-flow problem has an interesting interpretation. Suppose $y(v, w)$ are the dual decision variables corresponding to the first kind of con-

straints and $z(v)$ are the dual decision variables corresponding to the second kind of constraints.

$$\begin{array}{ll}
 \text{minimize:} & \sum_{(v,w) \in V \times W} c(v,w)y(v,w) \\
 \text{subject to:} & y(v,w) \geq z(v) - z(w) \quad \text{for all } v, w \in V \\
 & z(s) = 1 \\
 & z(t) = 0 \\
 & y(v,w) \geq 0 \quad \text{for all } v, w \in V.
 \end{array}$$

Consider the graph G again. A directed path from the source s to a sink t is a sequence of edges of the form $(s, v_1), (v_1, v_2), \dots, (v_{k-1}, v_k), (v_k, t)$ in E . A subset of edges $S \subseteq E$ is called an s-t cut if after deleting the edges in S there is no directed path from s to t . Every s-t cut provides a feasible solution to the dual problem as follows: We set $y(v, w) = 1$ if (v, w) is in the cut, 0 otherwise. We set $z(v)$ equal 1 if there is a path from s to v after making the cut S , 0 otherwise. One can check that the constraints of the dual linear program are indeed satisfied by such a solution.

In fact, a much stronger statement is true. The optimal solution to the dual problem is of this form i.e. it comes from an s-t cut. The proof of this statement is beyond the scope of this class. The size of an s-t cut is the sum of capacities of the edges in the cut. And thus the dual problem is equivalent to finding a min-cut in the graph G .

Chapter 16

Integer Programming

An integer program is a linear program in which some or all of the variables are required to be integers. The most common constraint is usually that some variable is required to be either 0 or 1. Even though integer programs are slight generalizations of linear programs, they are much harder to solve. It can be shown that solving a general integer program is an NP-hard problem i.e. we do not expect to there be any efficient algorithm to solve them. The best we can hope for is to analyze some special kinds of integer programs or to find approximation algorithm for the general case.

One naive approach to solve an integer program is to simply forget the integrality condition and solve the underlying linear program. This technique is called LP-relaxation. For nice integer programs, we can expect this method to provide a good approximate solution. However, this method can also provide really incorrect answers. Consider the following integer program

$$\begin{array}{ll} \text{maximize:} & x \\ \text{subject to:} & y \leq \frac{x}{n-0.5} \\ & x, y \geq 0 \\ & x, y \text{ are integers,} \end{array}$$

where n is a fixed integer. One can check that the feasible region for the LP-relaxation is a triangle with vertices $(0, 0)$, $(1, 0)$ and $(n - 0.5, 1)$ and the only integer points in this triangle are $(0, 0)$ and $(1, 0)$. So, the optimal solution to the LP-relaxation is $(n - 0.5, 1)$ but the optimal solution to the original integer program is $(1, 0)$. Hence, as n increases the LP solution diverges away from the IP solution.

16.1 Branch and Bound

One technique for fixing the issues with LP-relaxation is called branch-and-bound. In branch-and-bound we solve the LP-relaxation if we end with a non-integer solution, say with $x = k$ then we make two cases $x \leq \lfloor k \rfloor$ and $x \geq \lceil k \rceil$. The goal is to shrink the feasible region so that all the vertices have integer coordinates. We repeat the process for each case, essentially performing a DFS until we find integer solutions. This is best demonstrated with an example.

Consider the integer program:

$$\begin{array}{llll} \text{maximize:} & 4x & + & 5y \\ \text{subject to:} & x & + & y \leq 10 \\ & 3x & - & 4y \leq 6 \\ & x & , & y \geq 0 \\ & x, y & & \text{are integers.} \end{array} \quad (16.1)$$

Solving the LP-relaxation gives us the solution $(x, y) = (4, 1.5)$. As the y -coordinates is not an integer, we make two cases $y \leq 1$ and $y \geq 2$ and analyze each separately by adding it as an extra constraint.

$$\begin{array}{llll} \text{maximize:} & 4x & + & 5y \\ \text{subject to:} & x & + & y \leq 10 \\ & 3x & - & 4y \leq 6 \\ & x & , & y \geq 0 \\ & x, y & & \text{are integers.} \\ & & & y \leq 1 \end{array}$$

In this case, the optimal solution to the LP-relaxation is $(x, y) = (3.342, 1)$. This time x is not an integer, so we make further two cases $x \leq 3$ and $x \geq 4$. In the first case, we get

$$\begin{array}{llll} \text{maximize:} & 4x & + & 5y \\ \text{subject to:} & x & + & y \leq 10 \\ & 3x & - & 4y \leq 6 \\ & x & , & y \geq 0 \\ & x, y & & \text{are integers.} \\ & & & y \leq 1 \\ & x & & \leq 3, \end{array}$$

which has the optimal solution $(x, y) = (3, 1)$. In the second case, we get

$$\begin{array}{llll} \text{maximize:} & 4x & + & 5y \\ \text{subject to:} & x & + & y \leq 10 \\ & 3x & - & 4y \leq 6 \\ & x & , & y \geq 0 \\ & x, y & & \text{are integers.} \\ & & & y \leq 1 \\ & x & & \geq 4, \end{array}$$

the LP-relaxation and hence the integer program is infeasible. Going back we are left to analyze the case $y \geq 2$.

$$\begin{array}{llllll} \text{maximize:} & 4x & + & 5y & & \\ \text{subject to:} & x & + & y & \leq & 10 \\ & 3x & - & 4y & \leq & 6 \\ & x & , & y & \geq & 0 \\ & x, y & & & & \text{are integers.} \\ & & & y & \geq & 2. \end{array}$$

This has an optimal solution $(x, y) = (2, 2)$.

Thus we have two possible candidates for optimal solutions to the integer program: $(3, 1)$ and $(2, 2)$. We check the objective value $4x + 3y$ at each of these to conclude that $(2, 2)$ is indeed the optimal solution with objective value 18.

References

Bibliography

- Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, 2004. doi: 10.1017/CBO9780511804441.
- G  rard Cornu  jols, Javier Pe  a, and Reha T  t  nc  . Optimization Methods in Finance. Cambridge University Press, 2 edition, 2018. doi: 10.1017/9781107297340. URL <https://www.andrew.cmu.edu/user/gc0v/webpub/book.pdf>.
- M.Z. Proof of strong duality via farkas lemma. Mathematics Stack Exchange. URL <https://math.stackexchange.com/q/2474797>. URL:<https://math.stackexchange.com/q/2474797> (version: 2017-10-17).
- user2020. Question about kkt conditions and strong duality. Mathematics Stack Exchange. URL <https://math.stackexchange.com/q/3616646>. URL:<https://math.stackexchange.com/q/3616646> (version: 2020-04-09).
- Robert J. Vanderbei. Linear Programming. Springer New York, NY, 2014. ISBN 978-1-4614-7630-6. doi: 10.1007/978-1-4614-7630-6.