Linear Programming Theory and Algorithms

Sirong Luo

Faculty of Statistics and Data Science
Shanghai University of Finance and Economics

Table of Contents

Linear Programming

A linear program is an optimization problem whose objective is to minimize or maximize a linear function subject to a finite set of linear equality and linear inequality constraints.

Model

$$\begin{aligned} \min_{x} \quad & c^{\top} x \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{D}\mathbf{x} \geq \mathrm{d} \end{aligned}$$

For some vectors and matrices $\mathbf{c} \in \mathbb{R}^n, \mathbf{b} \in \mathbb{R}^m, \mathbf{d} \in \mathbb{R}^p, \mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{D} \in \mathbb{R}^{p \times n}$. The terms linear programming model or linear optimization model are also used to refer to a linear program. We will use these terms interchangeably.

Linear Programming

The following two simplified portfolio construction examples illustrate the use of linear programming as a modeling tool.

Example

2.1 (Fund allocation) You would like to allocate $\$80,000$ among four mutual funds that have different expected returns as well as different weights in large-, medium- and small-capitalization stocks.

Capitalization	Fund 1	Fund 2	Fund 3	Fund 4
Large	50%	30%	25%	60%
Medium	30%	10%	40%	20%
Small	20%	60%	35%	20%
Exp. return	10%	15%	16%	8%

The allocation must contain at least 35% large-cap, 30% mid-cap, and 15% small-cap stocks. Find an acceptable allocation with the highest expected return assuming you are only allowed to hold long positions in the funds.

Linear Programming

Example

2.1 (Fund allocation) This problem can be formulated as the following linear programming model.

Variables:

$$x_i : \text{amount (in \$1000 s) invested in fund } i \text{ for } i = 1, \ldots, 4.$$

Objective:

$$\max 0.10x_1 + 0.15x_2 + 0.16x_3 + 0.08x_4.$$

Constraints:

$$\begin{aligned}
0.50x_1 + 0.30x_2 + 0.25x_3 + 0.60x_4 &\geq 0.35 * 80 && \text{(large-cap)} \\
0.30x_1 + 0.10x_2 + 0.40x_3 + 0.20x_4 &\geq 0.30 * 80 && \text{(mid-cap)} \\
0.20x_1 + 0.60x_2 + 0.35x_3 + 0.20x_4 &\geq 0.15 * 80 && \text{(small-cap)} \\
x_1 + x_2 + x_3 + x_4 &= 80 && \text{(money to allocate)} \\
x_1, \ldots, x_4 &\geq 0 && \text{(long-only positions).}
\end{aligned}$$

Example

2.2 (Bond allocation) A bond portfolio manager has $\$100,000$ to allocate to two different bonds: a corporate bond and a government bond. These bonds have the following yield, risk level, and maturity:

Bond	Yield	Risk level	Maturity
Corporate	4%	2	3 years
Government	3%	1	4 years

The portfolio manager would like to allocate the funds so that the average risk level of the portfolio is at most 1.5 and the average maturity is at most 3.6 years. Any amount not invested in the bonds will be kept in a cash account that is assumed to generate no interest and does not contribute to the average risk level or maturity. In other words, assume cash has zero yield, zero risk level, and zero maturity.
How should the manager allocate funds to the two bonds to maximize yield? Assume the portfolio can only include long positions.

Example

2.2 (Bond allocation) This problem can be formulated as the following linear programming model. Linear programming model for bond allocation
Variables: x_1, x_2 : amounts (in \$1000 s) invested in the corporate and government bonds respectively.
Objective:

$$\max 4x_1 + 3x_2$$

Constraints:

$$\begin{aligned} x_1 + x_2 &\leq 100 && \text{(total funds)} \\ \frac{2x_1 + x_2}{100} &\leq 1.5 && \text{(risk level)} \\ \frac{3x_1 + 4x_2}{100} &\leq 3.6 && \text{(maturity)} \\ x_1, x_2 &\geq 0 && \text{(long-only positions)} \end{aligned}$$

Example

2.2 (Bond allocation)

or equivalently

$$
\begin{array}{lll}
\max & 4x_1 + 3x_2 & \\
\text{s.t.} & & \\
x_1 + x_2 & \leq 100 & \text{(total funds)} \\
2x_1 + x_2 & \leq 150 & \text{(risk level)} \\
3x_1 + 4x_2 & \leq 360 & \text{(maturity)} \\
x_1, x_2 & \geq 0 \quad \text{(long-only positions).} &
\end{array}
$$

Linear Programming

The linear programming model in Example 2.1 can be written more concisely using matrix-vector notation as follows:

$$\begin{array}{ll} \max & \mathbf{r}^\top \mathbf{x} \\ \text{s.t.} & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{D}\mathbf{x} \geq \mathbf{d} \\ & \mathbf{x} \geq \mathbf{0}, \end{array}$$

where $\mathbf{r} = \begin{bmatrix} 0.10 \\ 0.15 \\ 0.16 \\ 0.08 \end{bmatrix}, \mathbf{D} = \begin{bmatrix} 0.5 & 0.3 & 0.25 & 0.6 \\ 0.3 & 0.1 & 0.4 & 0.2 \\ 0.2 & 0.6 & 0.35 & 0.2 \end{bmatrix}, \mathbf{d} = \begin{bmatrix} 28 \\ 24 \\ 12 \end{bmatrix},$

$\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{b} = 80.$

Linear Programming

Likewise, the linear programming model in Example 2.2 can be written as

$$\begin{aligned} \max \quad & \mathbf{r}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

for $\mathbf{r} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$, $\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 4 \end{bmatrix}$, and $\mathbf{b} = \begin{bmatrix} 100 \\ 150 \\ 360 \end{bmatrix}$.

Linear Programming

A linear programming model is in standard form if it is written as follows:

Model

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

The standard form is a kind of formatting convention that is used by some solvers. It is also particularly convenient to describe the most popular algorithms for solving linear programming, namely the simplex and interior-point methods.

Linear Programming

The standard form is not restrictive. Any linear program can be rewritten in standard form. In particular, inequality constraints (other than non-negativity) can be rewritten as equality constraints after the introduction of a so-called slack or surplus variable. For instance, the linear program from Example 2.2 can be written as

$$
\begin{array}{ll}
\max & 4x_1 + 3x_2 \\
\text{s.t.} & \\
x_1 + x_2 + x_3 & \\
2x_1 + x_2 + x_4 & = 150 \\
3x_1 + 4x_2 + x_5 & = 360 \\
x_1, x_2, x_3, x_4, x_5 & \geq 0.
\end{array}
$$

Linear Programming

More generally, a linear program of the form

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

can be rewritten as

Model

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{s} = \mathbf{b} \\ & \mathbf{x}, \mathbf{s} \geq \mathbf{0} \end{aligned}$$

Linear Programming

It can then be rewritten, using matrix notation, in the following standard form:

$$\begin{aligned} \min \quad & \begin{bmatrix} c \\ 0 \end{bmatrix}^\top \begin{bmatrix} x \\ s \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} \mathrm{A} & I \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = b \\ & \begin{bmatrix} x \\ s \end{bmatrix} \geq \mathbf{0} \end{aligned}$$

Unrestricted variables can be expressed as the difference of two new non-negative variables. For example, consider the linear program

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}. \end{aligned}$$

Linear Programming

The unrestricted variable $\mathbf{x}$ can be replaced by $\mathbf{u} - \mathbf{v}$ where $\mathbf{u}, \mathbf{v} \geq \mathbf{0}$. Hence the above linear program can be rewritten as

$$\begin{aligned} \min \quad & \mathbf{c}^\top (\mathbf{u} - \mathbf{v}) \\ \text{s.t.} \quad & \mathbf{A}(\mathbf{u} - \mathbf{v}) \leq \mathbf{b} \\ & \mathbf{u}, \mathbf{v} \geq \mathbf{0}. \end{aligned}$$

It can then be rewritten, using matrix notation, in the following standard form:

$$\begin{aligned} \min \quad & \begin{bmatrix} c \\ 0 \end{bmatrix}^\top \begin{bmatrix} x \\ s \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} \mathrm{A} & I \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = b \\ & \begin{bmatrix} x \\ s \end{bmatrix} \geq \mathbf{0} \end{aligned}$$

Linear Programming

Unrestricted variables can be expressed as the difference of two new non-negative variables. For example, consider the linear program

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}. \end{aligned}$$

The unrestricted variable $\mathbf{x}$ can be replaced by $\mathbf{u} - \mathbf{v}$ where $\mathbf{u}, \mathbf{v} \geq \mathbf{0}$. Hence the above linear program can be rewritten as

$$\begin{aligned} \min \quad & \mathbf{c}^\top (\mathbf{u} - \mathbf{v}) \\ \text{s.t.} \quad & \mathbf{A}(\mathbf{u} - \mathbf{v}) \leq \mathbf{b} \\ & \mathbf{u}, \mathbf{v} \geq \mathbf{0}. \end{aligned}$$

Graphical Interpretation of a Two-Variable Example

Banks need to consider regulations when determining their business strategy. In this section, we consider the Basel III regulations (Basel Committee on Banking Supervision, 2011). We present a simplified example following the paper of Pokutta and Schmaltz (2012). Consider a bank with total deposits D and loans L. The loans may default and the deposits are exposed to early withdrawal. The bank holds capital C in order to buffer against possible default losses on the loans, and it holds a liquidity reserve R to buffer against early withdrawals on the deposits. The balance sheet of the bank satisfies $L + R = D + C$. Normalizing the total assets to 1 , we have $R = 1 - L$ and $C = 1 - D$.

Graphical Interpretation of a Two-Variable Example

Basel III regulations require banks to satisfy four minimum ratio constraints in order to buffer against different types of risk:
Capital ratio: $\frac{C}{L} \geq r_1$
Leverage ratio: $C \geq r_2$
Liquidity coverage ratio: $\frac{R}{D} \geq r_3$
Net stable funding ratio: $\frac{\alpha D + C}{L} \geq r_4$,
where the ratios $r_1, r_2, r_3, r_4, \alpha$ are computed for each bank based on the riskiness of its loans and the likelihood of early withdrawals on deposits. To illustrate, consider a bank with
$r_1 = 0.3, r_2 = 0.1, r_3 = 0.25, r_4 = 0.7, \alpha = 0.3$. Expressing the four ratio constraints in terms of the variables D and L, we get

$$D + 0.3L \leq 1$$
$$D \leq 0.9$$
$$0.25D + L \leq 1$$
$$0.7D + 0.7L \leq 1.$$

Graphical Interpretation of a Two-Variable Example

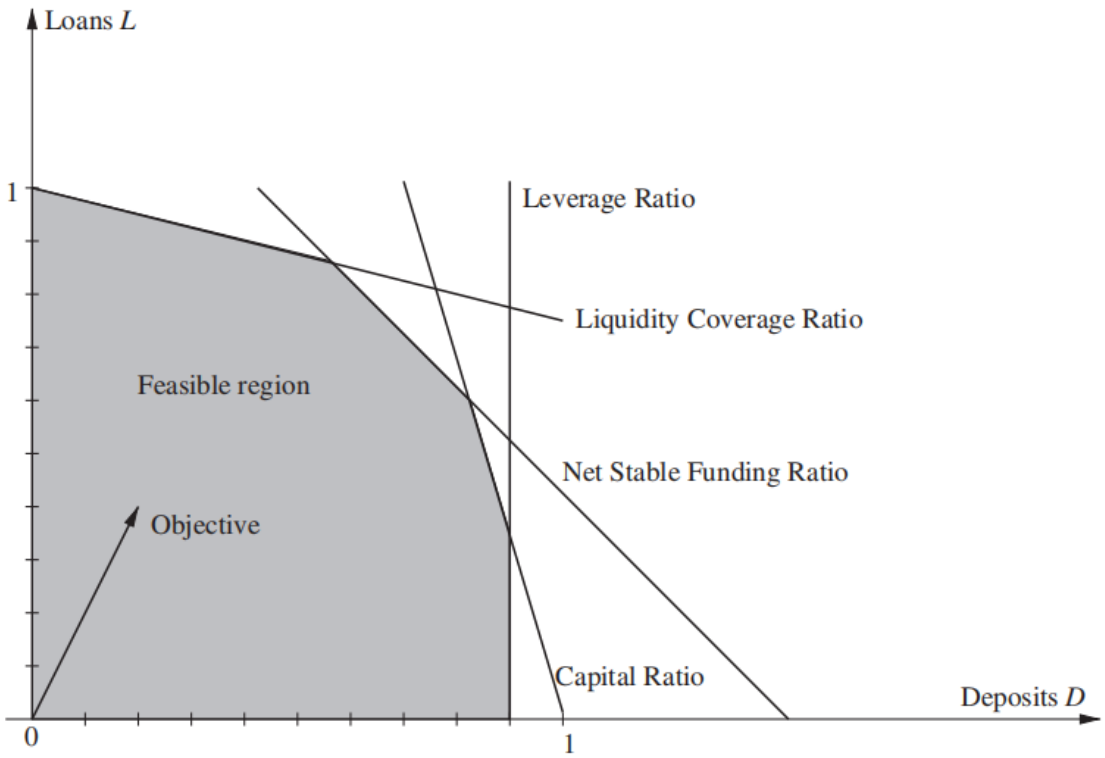

Figure: 2.1 Basel III regulations

Figure 2.1 displays a plot of the feasible region of this system of inequalities in the plane (D, L).

Graphical Interpretation of a Two-Variable Example

Given this feasible region, the objective of the bank is to maximize the margin income $m_D D + m_L L$ that it makes on its products; where m_D is the margin that the bank makes on its deposits and m_L is the margin charged on its loans.

For example, if $m_D = 0.02$ and $m_L = 0.03$, the best solution that satisfies all the constraints corresponds to the vertex $D = 0.571, L = 0.857$ on the boundary of the feasible region, at the intersection of the lines $0.25D + L = 1$ and $0.7D + 0.7L = 1$. This means that the bank should have 57.1% of its liabilities in deposits and 42.9% in capital, and it should have 85.7% of its assets in loans and the remaining 14.3% in liquidity reserve.

The fact that an optimal solution occurs at a vertex of the feasible region is a property of linear programs that extends to higher dimensions than 2: To find an optimal solution of a linear program, it suffices to restrict the search to vertices of the feasible region.

Numerical Linear Programming Solvers

There are a variety of both commercial and open-source software packages for linear programming. Most of these packages implement the algorithms described in Section 2.7 below. Next we illustrate two of these solvers by applying them to Example 2.1.

Excel Solver

Figure 2.2 displays a printout of an Excel spreadsheet implementation of the linear programming model for Example 2.1 as well as the dialog box obtained when we run the Excel add-in Solver. The spreadsheet model contains the three components of the linear program.

Numerical Linear Programming Solvers

The decision variables are in the range B4:E4. The objective is in cell F3. The left- and right-hand sides of the equality constraint are in the cells F4 and H4 respectively. Likewise, the left- and right-hand sides of the three inequality constraints are in the ranges F8:F10 and H8:H10 respectively. These components are specified in the Solver dialog box. In addition, the Solver options are used to indicate that this is a linear model and that the variables are non-negative.

Numerical Linear Programming Solvers

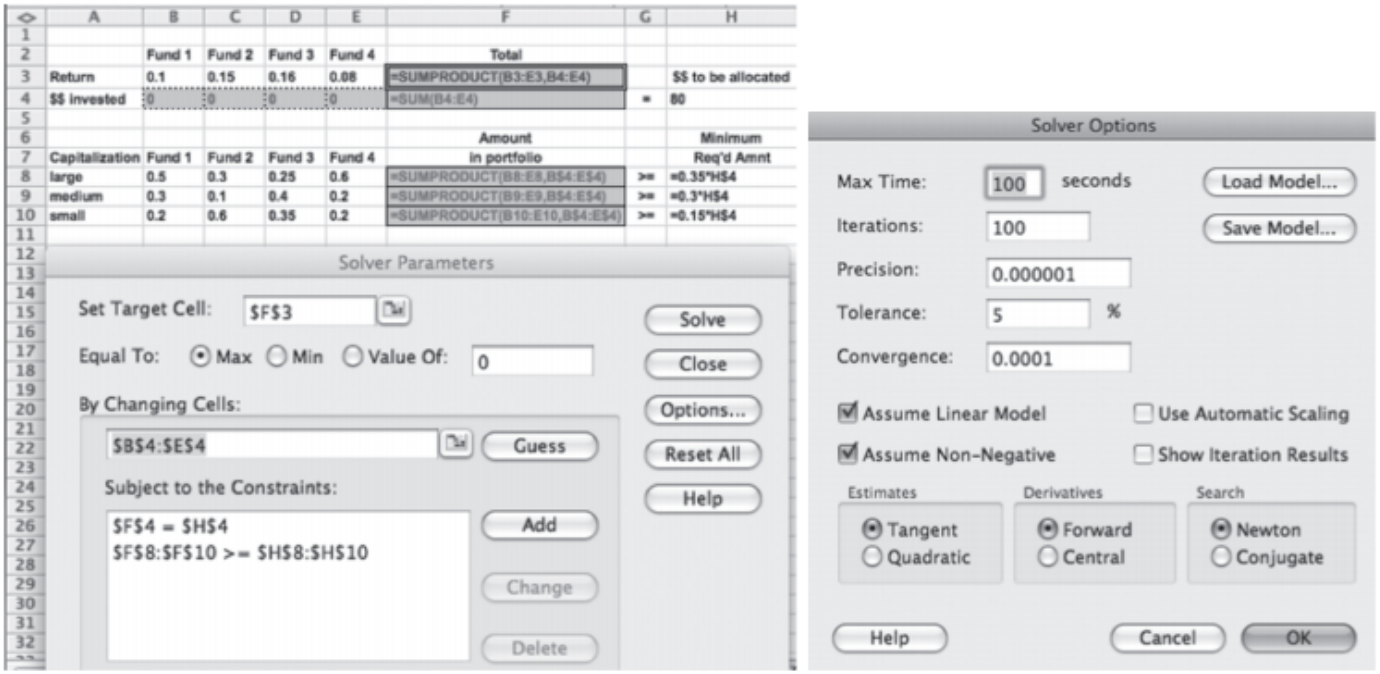

Figure: 2.2 Spreadsheet implementation and the Solver dialog box for the fund allocation model

Sensitivity Analysis

To make this information explicit in Excel Solver we request a sensitivity report after running it as shown in Figure 2.4

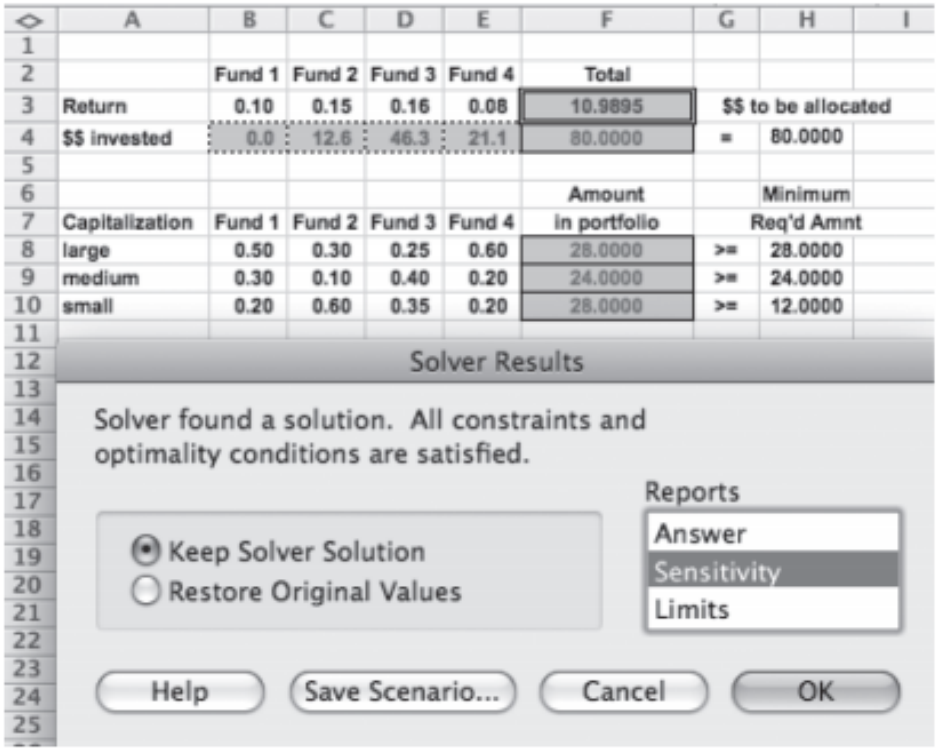

Figure: 2.4 Requesting sensitivity report in Solver

Sensitivity Analysis

Figure 2.5 displays the sensitivity report for Example 2.1.

Adjustable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
B4	$$ invested Fund 1	0	-0.00263158	0.1	0.00263158	1.00E+30
C4	$$ invested Fund 2	12.63157895	0	0.15	0.0166667	0.00142857
D4	$$ invested Fund 3	46.31578947	0	0.16	0.00166667	0.00625
E4	$$ invested Fund 4	21.05263158	0	0.08	0.01	0.00357143

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
F4	$$ invested Total	80.0000	0.22	80	21.0526	6.31579
F8	large in portfolio	28.0000	-0.231579	28	6	6.66667
F9	medium in portfolio	24.0000	-0.00526316	24	3.42857	14.6667
F10	small in portfolio	28.0000	0	12	16	1.00E+30

Figure: 2.5 Sensitivity report

Sensitivity Analysis

The values y_i^* can be found in the column labeled "Shadow Price". In addition, the "Allowable Increase" and "Allowable Decrease" columns indicate the range of change for each right-hand side of a constraint where the sensitivity analysis holds. For example, if the right-hand side of the large-capitalization constraint

$$0.5\mathbf{x}_1 + 0.3\mathbf{x}_2 + 0.25\mathrm{x}_3 + 0.6\mathrm{x}_4 \geq 28$$

changes from 28 to $28 + \Delta$, then the optimal value changes by $-0.231579 \cdot \Delta$. This holds provided Δ is within the allowable range [-6.6666,6]. If the requirement on large-cap stocks is reduced from 35% to 305 , the change in right-hand side is $\Delta = -0.05 * 80 = -4$, which is within the allowable range. Therefore the optimal objective value increases by $-0,231579 \cdot (-4) = 0.906316$. Because our units are in \$1000, this means that the expected return on an optimal portfolio would increase by \$926.32 if we relaxed the constraint on large-rap stocks by 5%, from 35% to 30%.

Sensitivity Analysis

The shadow prices of the non-negativity constraints are the "Reduced Cost" displayed in the initial part of the sensitivity report. This is also the convention for more general lower and upper bounds on the decision variables. Observe that in Example 2.1 the reduced costs of the non-zero variables are zero. The reduced costs also have a deeper meaning in the context of the simplex algorithm for linear programming as described in Section 2.7.1 below.

A linear programming model is non-degenerate if all of the allowable increase and allowable decrease limits are positive. The above linear programming model is non-degenerate.

Sensitivity Analysis

It is important to be mindful of this subtlety when interpreting the dual information. The ambiguity can be easily resolved by thinking in terms of sensitivity analysis.
In this particular example, it is clear that the shadow price of the first constraint should be non-negative as more capital should lead to a higher return. Likewise, it is clear that the shadow prices of the other constraints should be non-positive as more stringent diversification constraints, e.g., higher percentage in large cap,reduces the set of feasible portfolios and hence can only lead to portfolios with return less than or equal to the optimal return of the original problem.

Duality

Every linear program has an associated dual linear programming problem. The properties of these two linear programs and how they are related to each other have deep implications. In particular, duality enables us to answer the following kinds of questions:

- Can we recognize an optimal solution?
- Can we construct an algorithm to find an optimal solution?
- Can we assess how suboptimal a current feasible solution is?

Duality

There is a close connection between duality and sensitivity analysis. The vector of shadow prices of the constraints of a linear program corresponds precisely to the optimal solution of its dual.
Consider the following linear program in standard form, which we shall refer to as the primal problem:

Prime problem

$$\begin{aligned} \min \quad & \mathrm{c}^{\top}\mathrm{x} \\ \text{s.t.} \quad & \mathrm{Ax} = \mathrm{b} \\ & \mathrm{x} \geq 0 \end{aligned} \tag{2.1}$$

Duality

The following linear program is called the dual problem:

Dual problem

$$\begin{aligned} \max \quad & \mathbf{b}^{\mathrm{T}}\mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}^{\mathrm{T}}\mathbf{y} \leq \mathbf{c} \end{aligned} \tag{2.2}$$

Sometimes it is convenient to rewrite the constraints in the dual problem as equality constraints by means of slack variables. That is, problem (2.2) can also be written as

Dual problem

$$\begin{aligned} \max \quad & \mathbf{b}^{\top}\mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}^{\top}\mathbf{y} + \mathbf{s} = \mathbf{c} \\ & \mathbf{s} \geq \mathbf{0} \end{aligned} \tag{2.3}$$

There is a deep connection between the primal and dual problems. The next result follows by construction.

Duality

Theorem 2.3 (Weak duality)

Assume x is a feasible point for (2.1) and y is a feasible point for (2.2). Then

$$\mathbf{b}^{\top}\mathrm{y} \leq \mathrm{c}^{\mathrm{T}}\mathrm{x}$$

Proof Under the assumptions on x and y it follows, that

$$b^T y = (Ax)^T y = \left(A^T y\right)^T x \leq c^T x$$

Duality

Theorem 2.4 (Strong duality)

Assume one of the problems (2.1) or (2.2) is feasible. Then this problem is bounded if and only if the other one is feasible. In that case both problems have optimal solutions and their optimal values are the same.

Duality

Theorem 2.5

Assume $\mathrm{A} \in \mathbb{R}^{m \times n}$ and $\mathrm{b} \in \mathrm{R}^{m}$. In each of the following cases exactly one of the systems (I) or (II) has at solution but not both.

(a) Farkas's Lemma

$$Ax = b, x \geq 0 \quad \text{(I)}$$

$$A^{\top} y \leq 0, b^{\top} y < 0 \quad \text{(II)}$$

(b) Gordan's theorem

$$Ax = 0, x \geq 0 \quad \text{(I)}$$

$$A^{\top} y > 0 \quad \text{(II)}$$

(c) Stiemke's theorem

$$\mathrm{Ax} = 0, \mathrm{x} > 0 \quad \text{(I)}$$

$$\mathrm{A}^{\mathrm{T}} \mathrm{y} \geq 0 \quad \text{(II)}$$

Duality: Lagrangian function

We next present a derivation of the dual problem via the so-called Lagrangian function. This derivation has the advantage of introducing an important concept that we will encounter again later. For the optimization model:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_i(\mathbf{x}) = 0, \quad i = 1, \ldots, m \\ & h_j(\mathbf{x}) \leq 0, \quad j = 1, \ldots, p \end{aligned}$$

The Lagrangian function is:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f(\mathbf{x}) + \sum_{1}^{m} \lambda_i g_i(\mathbf{x}) + \sum_{1}^{p} \nu_j h_j(\mathbf{x}), \nu_j \geq 0$$

Associated with the optimization problem (2.1) consider the Lagrangian function defined by:

$$L(\mathrm{x}, \mathrm{y}, \mathrm{s}) := \mathrm{c}^{\mathrm{T}}\mathrm{x} + \mathrm{y}^{\mathrm{T}}(\mathrm{b} - \mathrm{A}\mathrm{x}) - \mathrm{s}^{\mathrm{T}}\mathrm{x}$$

Duality

The constraints of (2.1) can be encoded using the Lagrangian function via the following observation: For a given vector x

$$\max_{\substack{y,s \\ s\geq 0}} L(x,y,s) = \begin{cases} c^\top x & \text{if } \mathbf{Ax} = \mathbf{b} \text{ and } \mathrm{x} \geq 0 \\ +\infty & \text{otherwise} \end{cases}$$

Therefore the primal problem (2.1) can be written as

$$\min_{x} \max_{\substack{y\geq 0 \\ s\geq 0}} L(\mathbf{x},\mathbf{y},\mathbf{s})$$

On the other hand, observe that $L(\mathbf{x},\mathbf{y},\mathbf{s}) = \mathbf{b}^\top \mathbf{y} + \left(\mathbf{c} - \mathbf{A}^\top \mathbf{y} - \mathbf{s}\right)^\top \mathbf{x}$. Hence for a given pair of vectors $(\mathbf{y},\mathbf{s})$

$$\min_{\mathbf{x}} L(\mathbf{x},\mathbf{y},\mathbf{s}) = \begin{cases} \mathbf{b}^\top \mathbf{y} & \text{if } \mathbf{A}^\top \mathbf{y} + \mathbf{s} = \mathbf{c} \\ -\infty & \text{otherwise.} \end{cases} \tag{2.4}$$

Duality

The dual problem is obtained by flipping the order of the min and max operations in (2.4). Indeed, observe that the dual problem (2.3) can be written as

$$\max_{\substack{y,s \\ s\geq 0}} \min_{x} L(\mathbf{x}, \mathbf{y}, \mathbf{s})$$

A similar procedure can be applied to obtain the dual of a linear program that is not necessarily in standard form. For example, the primal problem

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{2.5}$$

can be written as

$$\min_{x} \max_{y\geq 0, s\geq 0} L(x, y, s)$$

for $L(\mathbf{x}, \mathbf{y}, \mathbf{s}) = \mathbf{c}^\top \mathbf{x} + \mathbf{y}^\top (\mathbf{b} - \mathbf{A}\mathbf{x}) - \mathbf{s}^\top \mathbf{x}$.

Duality

In this case the dual problem is

$$\max_{\mathbf{y}\geq 0,\ \mathrm{s}\geq 0} \min_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}, \mathbf{s})$$

and can be rewritten as

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{y} \leq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0} \end{aligned} \tag{2.6}$$

Duality

Again the weak and strong duality properties hold for the pair of problems (2.5) and (2.6). Consider the linear programming model of Example 2.1, namely

$$\begin{aligned} \max_{x} \quad & \mathbf{r}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{D}\mathbf{x} \geq \mathbf{d} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{2.7}$$

We give a derivation for its dual. Observe that (2.7) can be recast as

$$\max_{\mathbf{x}} \quad \min_{\substack{y,w,s \\ w \geq 0; s \geq 0}} \quad L(\mathbf{x}, \mathbf{y}, \mathbf{w}, \mathbf{s})$$

for

$$\begin{aligned} L(\mathbf{x}, \mathbf{y}, \mathbf{w}, \mathbf{s}) &= \mathbf{r}^\top \mathbf{x} + \mathbf{y}^\top (\mathbf{b} - \mathbf{A}\mathbf{x}) + \mathbf{w}^\top (\mathbf{D}\mathbf{x} - \mathbf{d}) + \mathbf{s}^\top \mathbf{x} \\ &= \mathbf{b}^\top \mathbf{y} - \mathbf{d}^\top \mathbf{w} + \mathbf{x}^\top \left(\mathbf{r} - \mathbf{A}^\top \mathbf{y} + \mathbf{D}^\top \mathbf{w} + \mathbf{s} \right) \end{aligned}$$

Duality

It follows that its dual $\min_{\substack{y,w,s \\ w\geq 0, s\geq 0}} \max_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{z})$ can be rewritten as

$$\begin{aligned} \min_{y,z} \quad & b^\top \mathbf{y} - d^\top \mathbf{w} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{y} - \mathbf{D}^\top \mathbf{w} \geq \mathbf{r} \\ & \mathbf{w} \geq \mathbf{0} \end{aligned} \tag{2.8}$$

An alternative way to obtain the dual (2.8) is to rewrite (2.7) in standard form and derive its standard dual. The latter turns out to be equivalent to (2.8). (See Exercise 2.6.)

Optimality Conditions

Consider again the linear programming problem (2.1). A powerful consequence of Theorem 2.4 is a set of optimality conditions that completely characterize the solutions to both (2.1) and (2.2).

Theorem 2.6 (Optimality conditions)

The vectors $\mathrm{x} \in \mathbb{R}^n$ and $(\mathbf{y}, \mathrm{s}) \in \mathbb{R}^m \times \mathbb{R}^n$ are respectively optimal solutions to (2.1) and (2.3) if and only if they satisfy the following system of equations and inequalities [KKT Conditions]:

$$\begin{aligned} \mathbf{A}^\top \mathbf{y} + \mathbf{s} &= \mathbf{c} \\ \mathbf{A}\mathbf{x} &= \mathbf{b} \\ \mathbf{x}, \mathbf{s} &\geq \mathbf{0} \\ x_i s_i &= 0, \quad i = 1, \ldots, n \end{aligned} \tag{2.9}$$

Optimality Conditions

The equations $x_i s_i = 0$ are known as the complementary slackness conditions. They imply that, if a dual constraint $\left(\mathbf{A}^\top \mathbf{y}\right)_i \leq c_i$ holds strictly (that is, $\left(\mathbf{A}^\top \mathbf{y}\right)_i < c_i$), then the corresponding primal variable x_i must be 0 . And conversely, if $x_i > 0$, the corresponding dual constraint is tight, that is, $\left(\mathbf{A}^\top \mathbf{y}\right)_i = c_i$.
The optimality conditions (2.9) provide an avenue for constructing algorithms to solve the linear programming problems (2.1) and (2.3). To lay the groundwork for discussing them, we next state two interesting results concerning the optimal solutions of a linear programming problem and its dual.

Optimality Conditions

Theorem 2.7 (Strictly complementary solutions)

Assume $\mathbf{A} \in \mathbb{R}^{m \times n}$ is full row rank, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^n$ are such that both (2.1) and (2.2) are feasible. Then there exist optimal solutions $\mathbf{x}^*$ to (2.1) and $(\mathbf{y}^*, \mathbf{s}^*)$ to (2.3) such that

$$\mathrm{x}^* + \mathrm{s}^* > 0$$

For a matrix $\mathbf{A}$ and a subset B of its columns, let $\mathbf{A}_B$ denote the submatrix of A containing the columns in B. For a square non-singular matrix $\mathbf{D}$, the notation $\mathbf{D}^{-\mathrm{T}}$ stands for $\left(\mathbf{D}^{-1}\right)^{\mathrm{T}}$.

Optimality Conditions

Theorem 2.8 (Optimal basic feasible solutions)

Assume $\mathbf{A} \in \mathbb{R}^{m \times n}$ is full row rank; $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^n$ are such that both (2.1) and (2.2) are feasible. Then there exists a partition $B \cup N = \{1, \ldots, n\}$ with $|B| = m$ and $\mathbf{A}_B$ non-singular, such that

$$\mathbf{x}_B^* = \mathbf{A}_B^{-1}\mathbf{b}, \quad \mathbf{x}_N^* = \mathbf{0}, \quad \mathbf{y}^* = \mathbf{A}_B^{-\top}\mathbf{c}_B$$

are optimal solutions to (2.1) and (2.2) respectively.

Algorithms for Linear Programming

We next sketch the two main algorithmic schemes for solving linear programs, namely **the simplex method** and **interior-point methods**. We follow the usual convention of assuming that the problem of interest is in standard form as in (2.1) and (2.3) and A has full row rank.

- The Simplex Method
- Dual Simplex Method
- Interior-Point Methods

The Simplex Method

One of the most popular algorithms for linear programming is the simpler method. It generates a sequence of iterates that satisfy $\mathrm{Ax} = \mathrm{b}, \mathrm{x} \geq 0,\ \mathrm{A}^{\top}\mathrm{y} + \mathrm{s} = \mathrm{c}$ and $x_{,} n_4 = 0$, with $i = 1 \ldots, n$. Each iteration of the algorithm aims to make progress towards satisfying $s \geq 0$. Theorem 2.6 guarantees that the algorithm terminates with an optimal solution when this goal is attained. The dual simplex method bs a variant that generates a sequence of iterates satisfying $\mathbf{Ax} = \mathbf{b}$, $A^{\top} y + s = c_{,} s \geq 0$, and $x_2 s_i = 0$. for $i = 1 \ldots \ldots n$. Each iteration of the algorithm aims to make progress towards satisfying $\mathbf{x} \geq 0$.

The simplex method relies on the property stated in Theorem 2.8. The gist of the method is to search for an optimal basis, that is, a subset $B \subseteq \{1, \ldots, n\}$ as in Theorem 2.8. To motivate and describe the algorithm we next introduce some terminology and key observations.

The Simplex Method

A basis is a subset $B \subseteq \{1, \ldots, n\}$ such that $|B| = m$ and A_B is a basis-$\mathrm{A}_{\mathrm{g}}^{-1}\ \mathrm{b}, \mathrm{x}_{\mathrm{N}} = 0$. Observe that x solves the system of equations $\mathrm{Ax} = \mathrm{b}$. The vector $\overrightarrow{\mathrm{x}}$ is a basic feasible solution if in addition $\hat{\mathrm{x}} \geq 0$. A basis B also defines the reduced cost $\bar{\mathbf{c}} = \mathbf{c} - \mathbf{A}^T \mathbf{A}_B^{-T} \mathbf{c}_B$.The following fact suggests the main idea for the simplex method.

Proposition 2.9 Assume $B \subseteq \{1, \ldots, n\}$ is a basis. Let $\hat{\mathbf{x}}$ and $\bar{\mathbf{c}}$ be respectively the corresponding basic solution and reduced cost vector. If $\bar{\mathbf{x}} \geq 0$ and $\bar{\mathbf{c}} \geq 0$ then $\bar{\mathrm{x}}$ is an optimal solution to (2.1). Furthermore, in this case $\bar{\mathbf{y}} = \mathbf{A}_B^{-\top} \bar{\mathbf{c}}_B$ is an optimal solution to (2.2).

The Simplex Method

An optimal basis is a basis that satisfies the condition $t \geq 0$ and $\epsilon \geq 0$ stated above. Given a basis B that is not optimal, the main idea of the simplex method is to generate a better basis by replacing an index from B. To that end, a possible avenue is as follows.To that end, a possible avenue is as follows. Suppose B is a basis with a basic feasible solution$\overline{\mathbf{X}}$. If B is not an optimal basis, then $\vec{c}_j < 0$ for some $j \notin B$. Thus for $\alpha > 0$ the point $\mathbf{x}(\alpha)$ defined by

$$\mathbf{x}_B(\alpha) = \overline{\mathbf{x}}_B - \alpha \mathbf{A}_B^{-1} \mathbf{A}_j,$$
$$x_j(\alpha) = \alpha, \quad x_i(\alpha) = 0 \text{ for all other indices } i \notin B \cup \{j\}$$

satisfies

$$\mathbf{c}^\top \mathbf{x}(\alpha) = \mathbf{c}^\top \overline{\mathbf{x}} + \alpha \bar{c}_j < \mathbf{c}^\top \overline{\mathbf{x}}$$

The Simplex Method

Hence we can get a point with better (lower) objective value than the current basic feasible solution $\bar{x}$. We would like this new point to remain feasible. Unless the problem is unbounded, there is a length $\alpha^* \geq 0$ that makes one of the current basic components ℓ of $\mathbf{x}$ drop to zero while keeping all of them non-negative. When $\alpha^* > 0$, a basis with a better basic feasible solution can be obtained by replacing ℓ with j.
The simplex method modifies the basis in this way, even in the degenerate case when $\alpha^* = 0$, which may occur in some iterations. Algorithm 2.1 gives a formal description of the simplex method.

Algorithm 2.1 The simplex method

1: start with a basis $B \subseteq \{1, \ldots, n\}$ such that $\bar{\mathbf{x}}$ is a basic feasible solution
2: **while** $\bar{\mathbf{c}} \not\geq \mathbf{0}$ **do**
3: choose an index j such that $\bar{c}_j < 0$
4: compute $\mathbf{u} = \mathbf{A}_B^{-1}\mathbf{A}_j$
5: **if** $\mathbf{u} \leq \mathbf{0}$ **then** HALT; the problem is unbounded **end if**
6: let $\alpha^* := \min_{i:u_i>0} \frac{\bar{x}_i}{u_i} = \frac{\bar{x}_\ell}{u_\ell}$
7: form a new basis by replacing ℓ with j
8: update the basic feasible solution by replacing $\bar{\mathbf{x}}$ with $\mathbf{x}(\alpha^*)$
9: **end while**

The Simplex Method

Observe that the basic feasible solution $\bar{\mathbf{x}}$ and the reduced cost $\bar{\mathbf{c}}$ corresponding to a basis B satisfy $\bar{\mathbf{x}}_N = \mathbf{0}$ and $\bar{\mathbf{c}}_B = \mathbf{0}$ where $N = \{1, \ldots, n\} \backslash B$. Hence the simplex method only needs to keep track of $\bar{\mathbf{x}}_B$ and $\bar{\mathbf{c}}_N$. We next illustrate the simplex method in the linear programming model from Example 2.2. If we start with the initial basis $B = \{3, 4, 5\}$ the algorithm proceeds as follows.

Iteration 1:

$B = \{3, 4, 5\}, \bar{\mathbf{x}}_B = \begin{bmatrix} \bar{x}_3 & \bar{x}_4 & \bar{x}_5 \end{bmatrix}^\top = \begin{bmatrix} 100 & 150 & 360 \end{bmatrix}^\top, \bar{\mathbf{c}}_N = \begin{bmatrix} \bar{c}_1 & \bar{c}_2 \end{bmatrix}^\top = \begin{bmatrix} -4 & -3 \end{bmatrix} \geqq \mathbf{0}$. Choose $j = 1$ as the new index to enter the basis. Compute $\mathbf{u} = \begin{bmatrix} u_3 & u_4 & u_5 \end{bmatrix} = \mathbf{A}_B^{-1} \mathbf{A}_j = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^\top$ and $\alpha^* := \min_{i: u_i > 0} \frac{\bar{x}_i}{u_i} = \frac{150}{2} = \frac{\bar{x}_4}{u_4}$. Hence $\ell = 4$ is the index leaving the basis. Update the basis and basic feasible solution to $B = \{3, 1, 5\}$ and $\bar{\mathbf{x}}_B = \begin{bmatrix} \bar{x}_3 & \bar{x}_1 & \bar{x}_5 \end{bmatrix}^\top = \begin{bmatrix} 25 & 75 & 135 \end{bmatrix}^\top$.

Iteration 2:

$B = \{3, 1, 5\}$, $\bar{\mathbf{x}}_B = \begin{bmatrix} \bar{x}_3 & \bar{x}_1 & \bar{x}_5 \end{bmatrix}^\top = \begin{bmatrix} 25 & 75 & 135 \end{bmatrix}^\top$, $\bar{\mathbf{c}}_N = \begin{bmatrix} \bar{c}_2 & \bar{c}_4 \end{bmatrix} = \begin{bmatrix} -1 & 0 \end{bmatrix}^\top \not\geq \mathbf{0}$. Choose $j = 2$ as the new index to enter the basis. Compute

$\mathbf{u} = \begin{bmatrix} u_3 & u_1 & u_5 \end{bmatrix}^\top = \mathbf{A}_B^{-1}\mathbf{A}_j = \begin{bmatrix} 1/2 & 1/2 & 5/2 \end{bmatrix}^\top$ and

$\alpha^* := \min_{i:u_i>0} \frac{\bar{x}_i}{u_i} = \frac{25}{1/2} = \frac{\bar{x}_3}{u_3}$. Hence $\ell = 3$ is the index leaving the basis. Update the basis and basic feasible solution to $B = \{2, 1, 5\}$ and

$\bar{x}_B = \begin{bmatrix} \bar{x}_2 & \bar{x}_1 & \bar{x}_5 \end{bmatrix}^\top = \begin{bmatrix} 50 & 50 & 10 \end{bmatrix}^\top$.

Iteration 3:

$B = \{2, 1, 5\}$, $\bar{x}_B = \begin{bmatrix} \bar{x}_2 & \bar{x}_1 & \bar{x}_5 \end{bmatrix}^\top = \begin{bmatrix} 50 & 50 & 10 \end{bmatrix}^\top$, $\bar{\mathbf{c}}_N = \begin{bmatrix} c_3 & c_4 \end{bmatrix}^\top = \begin{bmatrix} 2 & 1 \end{bmatrix}^\top \geq \mathbf{0}$. Hence B is an optimal basis and

$$\bar{\mathbf{x}} = \begin{bmatrix} 50 & 50 & 0 & 0 & 10 \end{bmatrix}^\top$$

is an optimal solution.

The Simplex Method

Notice how, geometrically, the simplex iterations move from one vertex of the feasible region to an adjacent vertex until an optimum solution is identified. See Figure 2.7.

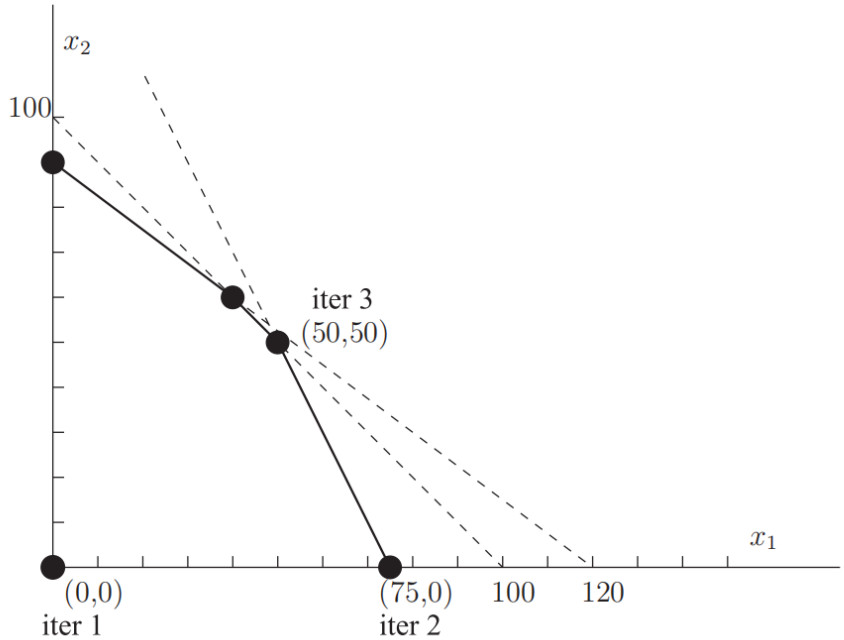

Figure: 2.7 Simplex iterations

Dual Simplex Method

The above version of the simplex method is a primal version that generates primal feasible iterates and aims for dual feasibility. The dual simplex method generates dual feasible iterates and aims for primal feasibility. The logic behind the algorithm is similar. Suppose B is a basis with reduced cost $\bar{\mathbf{c}} \geq \mathbf{0}$. This means that $\bar{\mathbf{y}} = \mathbf{A}_B^{-\top}\mathbf{c}_B$ is a dual feasible solution with slack $\bar{\mathbf{c}} = \mathbf{c} - \mathbf{A}^\top \mathbf{y} \geq \mathbf{0}$. If B is not an optimal basis, then $\bar{x}_\ell < 0$ for some $\ell \in B$. Let $\mathbf{e}_\ell \in \mathbb{R}^n$ denote the vector with ℓ th component equal to 1 and all others equal to 0 . For $\alpha > 0$ the vector $\mathbf{y}(\alpha)$ defined by

$$\mathbf{y}(\alpha) = \bar{\mathbf{y}} - \alpha \mathbf{A}_B^{-\top}\mathbf{e}_\ell$$

satisfies

$$\mathbf{b}^\top \mathbf{y}(\alpha) = \mathbf{b}^\top \bar{\mathbf{y}} - \alpha \bar{x}_\ell > \mathbf{b}^\top \bar{\mathbf{y}}.$$

Dual Simplex Method

Observe that the slack of $\mathbf{y}(\alpha)$ is

$$\mathbf{c}(\alpha) = \overline{\mathbf{c}} + \alpha \mathbf{A}^\top \mathbf{A}_B^{-\top} \mathbf{e}_\ell.$$

Hence we can get a point with better (lower) objective value than the current basic feasible solution $\overline{\mathrm{x}}$. Unless the problem is unbounded, there is a length $\alpha^* \geq 0$ that makes one of the non-basic components j of $\mathbf{c}(\alpha)$ drop to zero while keeping all of them non-negative. In this case a basis with a better dual solution can be obtained by replacing ℓ with j. Algorithm 2.2 gives a formal description of the dual simplex method. We illustrate the dual simplex in the following variation of Example 2.2. Suppose we add the constraint

$$6x_1 + 5x_2 \leq 500.$$

Dual Simplex Method

Algorithm 2.2 Dual simplex method

1: start with a basis $B \subseteq \{1, \ldots, n\}$ such that the reduced cost $\bar{\mathbf{c}}$ is non-negative
2: **while** $\bar{\mathbf{x}} \not\geq \mathbf{0}$ **do**
3: choose an index $\ell \in B$ such that $\bar{x}_\ell < 0$
4: compute $\mathbf{v} = \mathbf{A}^\top \mathbf{A}_B^{-\top} e_\ell$
5: **if** $\mathbf{v} \geq \mathbf{0}$ **then** HALT; the problem is unbounded **end if**
6: let $\alpha^* := \min_{i:v_i<0} \frac{\bar{c}_i}{|v_i|} = \frac{\bar{c}_j}{|v_j|}$
7: form a new basis by replacing ℓ with j
8: update the dual feasible solution by replacing $\bar{\mathbf{y}}$ with $\mathbf{y}(\alpha^*)$
9: **end while**

After adding the relevant new slack variable the new linear program is

$$
\begin{aligned}
\min \quad & -4x_1 - 3x_2 \\
\text{s.t.} \quad & \\
& x_1 + x_2 + x_3 = 100 \\
& 2x_1 + x_2 + x_4 = 150 \\
& 3x_1 + 4x_2 + x_5 = 360 \\
& 6x_1 + 5x_2 = 500 \\
& x_1, x_2, x_3, x_4, x_5, x_6 \geq 0
\end{aligned}
$$

Dual Simplex Method

If we start with the initial basis $B = \{1, 2, 5, 6\}$ the algorithm proceeds as follows. **Iteration 1:**
$B = \{1, 2, 5, 6\}, \bar{\mathbf{c}}_N = \begin{bmatrix} \bar{c}_3 & \bar{c}_4 \end{bmatrix}^\top = \begin{bmatrix} 2 & 1 \end{bmatrix}^\top \geq \mathbf{0}$, and

$$\bar{\mathbf{x}}_B = \begin{bmatrix} \bar{x}_1 & \bar{x}_2 & \bar{x}_5 & \bar{x}_6 \end{bmatrix}^\top = \begin{bmatrix} 50 & 50 & 10 & -50 \end{bmatrix}^\top \ngeq \mathbf{0}.$$

Choose $\ell = 6$ as the index to leave the basis. Compute $\mathbf{v} = \mathbf{A}^\top \mathbf{A}_B^{-\top} e_\ell = \begin{bmatrix} 0 & 0 & -4 & -1 & 0 & 1 \end{bmatrix}^\top$ and $\alpha^* := \min_{i:v_i<0} \frac{\bar{c}_i}{|v_i|} = \frac{2}{4} = \frac{\bar{c}_3}{|v_3|}$. Hence $j = 3$ is the index entering the basis.
Update the basis, reduced cost, and basic solution respectively to
$B = \{1, 2, 5, 3\}, \bar{c}_N = \begin{bmatrix} \bar{c}_4 & \bar{c}_6 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix}^\top$, and
$\bar{\mathbf{x}}_B = \begin{bmatrix} \bar{x}_1 & \bar{x}_2 & \bar{x}_5 & \bar{x}_3 \end{bmatrix}^\top = \begin{bmatrix} 62.5 & 25 & 72.5 & 12.5 \end{bmatrix}^\top$. The new basic solution is non-negative and hence it is an optimal solution.

Interior-Point Methods

In contrast to the simplex method, interior-point methods generate a sequence of iterates that satisfy $\mathbf{x}, \mathbf{s} > \mathbf{0}$. Each iteration of the algorithm aims to make progress towards satisfying $\mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{A}^\top \mathbf{y} + \mathbf{s} = \mathbf{c}$, and $x_i s_i = 0, i = 1, \ldots, n$.

Throughout this section we use the following notational convention: Given a vector $\mathbf{x} \in \mathbb{R}^n$, let $\mathbf{X} \in \mathbb{R}^{n \times n}$ denote the diagonal matrix defined by $X_{ii} = x_i, i = 1, \ldots, n$, and let $\mathbf{1} \in \mathbb{R}^n$ denote the vector whose components are all ones.

The optimality conditions (2.9) can be restated as

$$\begin{bmatrix} \mathbf{A}^\top \mathbf{y} + \mathbf{s} - \mathbf{c} \\ \mathbf{A}\mathbf{x} - \mathbf{b} - \mathbf{b} \\ \mathbf{X}\mathbf{S}\mathbf{1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathrm{x},\ \mathrm{s} \geq \mathbf{0}$$

Interior-Point Methods

Given $\mu > 0$, let $(\mathbf{x}(\mu), \mathbf{y}(\mu), \mathbf{s}(\mu))$ be the solution to the following perturbed version of the above optimality conditions:

$$\begin{bmatrix} \mathbf{A}^\top \mathbf{y} + \mathbf{s} - \mathbf{c} \\ \mathbf{A}\mathbf{x} - \mathbf{b} - \mathbf{b} \\ \mathbf{X}\mathbf{S}\mathbf{1} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mu \mathbf{1} \end{bmatrix}, \quad \mathbf{x}, \mathbf{s} > \mathbf{0}.$$

The first condition above can be written as $\mathbf{r}_\mu(\mathbf{x}, \mathbf{y}, \mathbf{s}) = \mathbf{0}$ for the residual vector

$$\mathbf{r}_\mu(\mathbf{x}, \mathbf{y}, \mathbf{s}) := \begin{bmatrix} \mathbf{A}^\top \mathbf{y} + \mathbf{s} - \mathbf{c} \\ \mathbf{A}\mathbf{x} - \mathbf{b} \\ \mathbf{X}\mathbf{S}\mathbf{1} - \mu \mathbf{1} \end{bmatrix}.$$

Interior-Point Methods

The central path is the set $\{(\mathbf{x}(\mu), \mathbf{y}(\mu), \mathbf{s}(\mu)) : \mu > 0\}$. It is intuitively clear that $(\mathbf{x}(\mu), \mathbf{y}(\mu), \mathbf{s}(\mu))$ converges to an optimal solution to both (2.1) and (2.3) as μ goes to 0 .

This suggests the following algorithmic strategy: Suppose $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ is "near" $(\mathbf{x}(\mu), \mathbf{y}(\mu), \mathbf{s}(\mu))$ for some $\mu > 0$. Use $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ to move to a better point $(\mathbf{x}^+, \mathbf{y}^+, \mathbf{s}^+)$"near" $(\mathbf{x}(\mu^+), \mathbf{y}(\mu^+), \mathbf{s}(\mu^+))$for some $\mu^+ < \mu$. It can be shown that if a point $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ is on the central path, then the corresponding value of μ satisfies $\mathbf{x}^\top \mathbf{s} = n\mu$. Likewise, given $\mathbf{x}, \mathbf{s} > \mathbf{0}$, define

$$\mu(\mathbf{x}, \mathbf{s}) := \frac{\mathbf{x}^\top \mathbf{s}}{n}$$

Interior-Point Methods

To move from a current point $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ to a new point, we use the so-called Newton step; that is, the solution to the following system of equations:

$$\begin{bmatrix} \mathbf{0} & \mathbf{A}^\top & \mathbf{I} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{S} & \mathbf{0} & \mathbf{X} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{s} \end{bmatrix} = \begin{bmatrix} \mathbf{c} - \mathbf{A}^\top\mathbf{y} - \mathbf{s} \\ \mathbf{b} - \mathbf{A}\mathbf{x} \\ \mu\mathbf{1} - \mathbf{X}\mathbf{S}\mathbf{1} \end{bmatrix} \tag{2.10}$$

Algorithm 2.3 presents a template for an interior-point method.

Algorithm 2.3 Interior-point method

1: choose $\mathbf{x}^0, \mathbf{s}^0 > 0$
2: **for** $k = 0, 1, \ldots$ **do**
3: solve the Newton system (2.10) for $(\mathbf{x}, \mathbf{y}, \mathbf{s}) = (\mathbf{x}^k, \mathbf{y}^k, \mathbf{s}^k)$ and $\mu := 0.1\mu(\mathbf{x}^k, \mathbf{s}^k)$
4: choose a step length $\alpha \in (0, 1]$ and set $(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}, \mathbf{s}^{k+1}) = (\mathbf{x}^k, \mathbf{y}^k, \mathbf{s}^k) + \alpha(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$
5: **end for**

Interior-Point Methods

The step length α in step 4 should be chosen so that $\mathrm{x}^{k+1},\ \mathrm{s}^{k+1} > 0$ and the size of $\mathbf{r}_\mu\left(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}, \mathbf{s}^{k+1}\right)$ is sufficiently smaller than $\mathbf{r}_\mu\left(\mathbf{x}^{k}, \mathbf{y}^{k}, \mathbf{s}^{k}\right)$. A linesearch procedure as the one described in Algorithm 2.4 is a popular strategy for choosing the step length α.

Algorithm 2.4 Line search to select the step length α

1: let $\alpha_{\max} := \max\{\alpha : (\mathbf{x}^k, \mathbf{y}^k, \mathbf{s}^k) + \alpha(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s}) \geq 0\}$
2: start with $\alpha := 0.99\alpha_{\max}$
3: **while** $\|\mathbf{r}_\mu((\mathbf{x}^k, \mathbf{y}^k, \mathbf{s}^k) + \alpha(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s}))\| \geq (1 - 0.01\alpha)\|\mathbf{r}_\mu(\mathbf{x}^k, \mathbf{y}^k, \mathbf{s}^k)\|$ **do**
4: $\quad\alpha := \alpha/2$
5: **end while**

Interior-Point Methods

In contrast to the primal and dual simplex methods, which in principle generate either primal or dual feasible iterates and terminate after finitely many iterations, interior-point methods typically generate infeasible iterates and converge to the optimal solution in the limit.

In practice, the convergence is so fast that in a few iterations the algorithm yields iterates that are within machine precision of an exact optimal solution. The algorithm can also be enhanced to detect infeasibility. It relies on the fact that when the primal or dual problem is infeasible, the norm of the residual r(x, y, s) cannot be driven to zero.

Interior-Point Methods

When applied to Example 2.2 starting from $x^0 = \mathbf{s}^0 = \begin{bmatrix} 100 & \cdots & 100 \end{bmatrix}^\top$, $\mathbf{y}^0 = \mathbf{0}$, the above interior-point algorithm generates the following sequence of iterates. (For ease of notation we only display the first two entries of each iterate.)

Iteration	0	1	2	$\cdots$	8	9
x_1	100	19.7084	17.2976	$\cdots$	49.9383	49.9962
x_2	100	57.3930	41.6795	$\cdots$	50.0436	50.0011