MA231

Operational Research Methods Lecture Notes[1]

2018/19

Dr Giacomo Zambelli

THE LONDON SCHOOL OF ECONOMICS AND POLITICAL SCIENCE

[1]Notes are based in part on previous notes by Gautam Appa, Katerina Papadaki, Susan Powell, Alan Pryor, Gregory Sorkin, László Végh, and on the textbook [HL].

Contents

Chapter 1

Introduction

Operational Research is the discipline of applying advanced analytical methods to making better decisions; in non-British usage it is known as *Operations Research*, and in either case it is commonly written just as *OR*. A slightly different definition describes OR as the science of efficient allocation and usage of scarce resources. The term *Management Science* is sometimes used as a synonym of OR, and sometimes as a broader discipline also incorporating *Decision Science*.

OR is a modern discipline, born in the Second World War era (see the historical background in the next section). The need for such a discipline arose from the appearance of large scale industrial production triggered by the industrial revolution, and the increasing complexity of organisational and logistical tasks. It became impossible for a single mind to grasp every detail of the processes in large organisations, and come up with the best decision, or even to identify what the possible options are. OR provides systematical modelling techniques for production, logistics and organisational processes, and for application of scientific methods to enhance decision making. The development of the field was fostered by the computer revolution in the mid-twentieth century, which provided the main technical tool of OR, enabling the solution of large-scale optimisation problems.

OR comprises a broad range of theoretical methods and application areas, the latter including e.g. manufacturing, transportation, telecommunication, health care, public policy, marketing, revenue management, construction; see some examples on the lecture slides. A key step in applications is *modelling*: constructing a mathematical representation of the real-world problem that accurately represents the essential features. A general scheme of the overall process is the following (see [HL], Chapter 2):

1. Define the problem and gather relevant data.
2. Formulate a mathematical model to represent the problem.
3. Develop a computer-based procedure for deriving solutions to the problem from the model.
4. Test the model and refine it as needed.
5. Prepare for the ongoing application of the model.
6. Implement.
7. Maintain and continue refining.

The course MA231 focuses on the core steps 2 and 3 above, and comprises three parts:

- Part I focuses on Mathematical Optimisation, with particular emphasis on Linear Programming but also covering other topics.
  - *Linear Programming (LP)*, a mathematical optimisation method that is the most important general tool of OR.
  - *Network models*: Network flow, shortest paths, Project Management and Critical Paths,
  - *Dynamic Programming*, a general algorithmic technique for making a sequence of interrelated decisions.
- Part II is dedicated to a variety of models and tools arising in applications.
  - *Game Theory*, a general theory of rational agents' behaviour under competition.
  - *Markov Chains*, an abstract probabilistic model arising in a broad range of applications.
  - *Inventory Models*, which pertains the optimal management of stock levels.
  - *Queuing Theory*, the mathematical study of the long-term behaviour of queues.
- Part III is on Simulation and Optimisation modeling. This part of the course focuses on modeling and implementation
  - *Mathematical Optimisation*: we will revisit mathematical optimisation, in particular Linear Programming, but from the perspective of using optimisation in applications and of implementing and solving mathematical optimisation models using Excel Solver.
  - *Simulation*, a general method to understand stochastic (random) system behaviour by simulating events.

The course will be mainly based on the following textbook:

[HL] Hillier, F. S., Lieberman, G. J. *Introduction to Operations Research.* McGraw-Hill, 7th edition, 2005 / 9th edition, 2010.

Copies of both editions are available from the library; differences between the two editions in the relevant chapters are minor. The book does not cover some parts of the topics covered in these notes. The lecture notes contain sufficient detail to prepare you for the exam. Still, the book is recommended as it contains much more detail, context, case studies, and examples.

1.1 Historical background

While some methods, results, and ideas originate from mathematics of the preceding centuries, modern operational research was only established in the Second World War era. The massive logistic and organisational questions arising in the planning of military operations engaged the efforts of scientists from various fields, including several outstanding mathematicians. This new field emerged in multiple scientific communities simultaneously, in the UK, the US, and the Soviet Union.

The term 'Operational' Research was coined by physicist *Albert P. Rowe* in 1937 to describe work on the usage and integration of radars into the air defence system, as opposed to 'technical' research on improvement of radar technology. Application of mathematical methods sometimes resulted in policies that contradicted military common sense, but that turned out to be highly successful when implemented.

An eminent example is a project led by *Patrick Blackett* in 1943. The Germans had discovered how to listen to British radar signals, allowing them to warn their U-boats (submarines) of coming attacks against them, giving the boats enough time to dive. As a consequence, the number of successful Allied sinkings of U-boats drastically decreased. Among other suggestions, the Allied OR team proposed to concentrate attacks in certain areas, forcing the U-boats to dive so frequently that they would exhaust their air and fuel supplies and would eventually have to face an attack on the surface. The plans got implemented despite strong opposition from military commanders, and increased the fraction of attacked U-boats successfully sunk from 2% to 45%.

The Allies' OR team also considered the defense of convoys of ships. Given constraints on the availability of air cover, and warships to defend the perimeter, is it better to organise convoys and their defence with a large number of small convoys or a small number of large convoys? What principle might be used in answering this question? (*Hint:* consider the convoy as a square, with defence provided by warships around the perimeter and long range airplanes.)

The most fundamental tool of OR, *Linear Programming (LP)*, was developed in the Second World War era by George Dantzig and John von Neumann in the US, and by Leonid Kantorovich in the USSR. In 1947 Dantzig invented the Simplex Method for solving linear programmes, and it remains the most heavily used optimisation algorithm.

After the war, the industrial boom vastly increased the size of corporations and the volume of production. This triggered a demand for systematic modelling and decision-making tools to make better predictions and to increase efficiency. For this reason, OR developed rapidly and has played a crucial role in all fields of industry ever since.

Suggested reading

- Main reading: Chapters 1 and 2 of [HL] (both editions).
- History of OR: A bibliography can be found on the INFORMS Society's website, here: https://www.informs.org/Explore/History-of-O.R.-Excellence/Bibliographies/The-Origins-of-OR.
- Applications: The INFORMS Society's *Franz Edelman Award* recognises successful applications of OR and analytics techniques. Descriptions of current and past winners and finalists can be found here: https://www.informs.org/Recognizing-Excellence/INFORMS-Prizes/Franz-Edelman-Award. Some recent winners were:

  2017: Holiday Retirement for its successful introduction of revenue management.

  2016: UPS for its proprietary routing software *ORION (On-Road Integrated Optimization and Navigation)*.

  2015: Syngenta for *Good Growth through Advanced Analytics*, applying OR methods to make better breeding decisions that reduce the time and cost required to develop crops with higher productivity.

  2014: The US Center for Disease Control and Prevention (CDC) for *Polio Eradicators Use Integrated Analytical Models to Make Better Decisions.*

- More applications: A website aimed at corporate executives, `http://www.scienceofbetter.org`, also includes a large number of OR application success stories.

Paired Kidney Exchange

In class, you will discuss a medical application of OR. The project was also one of the finalists of the Edelman Award in 2014. As preparation, first watch a 15 min talk by the Nobel prize laureate *Al Roth*, one of the main contributors on the problem. You can access the video link from Moodle. After that, read the article (handed out, also available on Moodle).

> P. Ferrari, M. De Klerk. Paired kidney donations to expand the living donor pool. *Journal of Nephrology*, 22(6): 699-707 (2009)

The following questions are meant to facilitate your understanding.

1. What does *paired kidney exchange* mean?
2. Why do the operations need to be carried out simultaneously?
3. What is a three-way exchange, and why is it problematic?
4. How do blood types affect compatibility, and what is the problematic issue related to blood type O?
5. What is the main advantage of altruistic donor chains?
6. What are the legal issues related to paired kidney exchange?

If you found the article interesting, you are encouraged to read one of the referenced papers, and prepare a 5 minute presentation to the class summarising the content.

1.2 Exercises

The course assumes familiarity with the following basic concepts: *partial derivatives, convexity, minimising functions; solving a set of linear equations using Gaussian elimination; infinite series; random variables: expectation, variance, z-values, common distributions.*

The first three simple exercises require some of the concepts mentioned above. If you have difficulties with them, please refer to MA107/MA100 and ST107/ST102 lecture notes. If you still have difficulties, ask your class teacher for help with the concepts, during office hours or in class.

Exercise 1.1. Consider the function

$$f(x) = \frac{a}{x} + bx$$

on the domain $x > 0$, for some parameters a and b.

(a) For what values of a and b will the function be convex? Concave?

(b) Compute the minimum and maximum values of the function on $x \geq 0$, depending on the parameter values.

Exercise 1.2. Solve the following system of linear equations:

$$\begin{pmatrix} 0 & 2 & 1 \\ 1 & -2 & -3 \\ -1 & 1 & 2 \end{pmatrix} \cdot x = \begin{pmatrix} -8 \\ 0 \\ 3 \end{pmatrix}$$

Exercise 1.3. The number of microwave ovens sold by a retailer on a day follows a normal distribution with mean 45 and variance 6. What is the smallest inventory the shop needs to keep so that all customers can be served with at least 95% probability?

Exercise 1.4. (a) We roll a die n times. What is the probability that we get a 6 in the n^{th} roll but never before? (b) What is the expected number of times we have to roll a die to get a 6?

Part I

Mathematical Optimisation

Chapter 2

Examples of linear programs

The term *Mathematical Optimisation* refers to using **mathematical** tools for **optimally** allocating and using limited resources when planning activities. Mathematical Optimisation deals therefore with optimisation problems. An optimisation problem consists of maximising or minimising some function – representing, for example, total profit or cost, or total number of staff, time, or other resources needed to perform certain tasks – subject to a number of constraints, representing limitations on the resources or on the way these can be used.

The focus of the following sections is on *Linear Programming (LP)*, a simple, yet fundamental mathematical programming model. The emphasis in our course is on understanding the simple structure of LP through examples and diagrams and motivating the study through applications. There are literally hundreds of practical applications of LP in a wide variety of fields ranging from logistics problems in industry, financial markets, natural and social sciences, and many more. The model and its applications started around the Second World War, with Leonid Kantorovich using it to model centrally-managed Soviet economy and George Dantzig in the States to model logistics problems arising from the war effort and the subsequent industrial developments. Dantzig also devised the first effective algorithm for solving linear programming problems – the so-called Simplex Algorithm – which to this day is implemented in all commercial linear programming solvers. The advent of computers and the exponential increase in computing power allows us today to efficiently solve linear programming problems with up to a few million variables.

The material of the lectures on linear programming is covered in Chapters 3-9 in [HL].

The purpose of this lecture is to introduce the type of problems that can be represented as linear programs. We start by giving two examples before discussing the general form of a linear program.

2.1 Introductory examples

Example 1 - Computer manufacturing

A computer manufacturer produces 5 families of laptops. The company has to cope with shortages from suppliers of two components: solid state drives (SSDs) and memory boards. The next table illustrates the 5 types of computers and the use of components.

System	L1	L2	L3	L4	L5
Price (£)	2000	1400	1000	800	500
# SSDs	1	.7	0	.3	0
# Memory boards	6	4	4	2	2

For example, 7 out of 10 of the family L2 of laptops use SSDs (the remaining 3 use regular hard-disks), and the average price of a model in the family L2 is £1400. The following difficulties are anticipated for the next quarter:

- The supplier of CPUs can provide at most 8000 units.
- The supplier of SSDs can provide at most 3000 units.
- The supplier of memory boards can provide at most 14000 units.

A demand of 5000 units is estimated for the first two types of laptops, and a demand of 4000 for the last three types. Furthermore, there are already 700 orders of laptops L2 and L5.

The company would like to devise a production plan for the next quarter in order to maximise their profit.

1. *What is the manufacturer's objective?* To maximise his profit.
2. *What restricts his profit?* The number of CPUs, SSDs, and memory boards.
3. *What can the company decide to do?* The number of each type of laptop to produce.

In words, the problem is to

maximise	profit from producing computers
subject to	number of components used does not exceed the availability;
	production does not exceed estimated demand;
	orders already placed are satisfied.

It is necessary to decide on the units:

- Profit is measured in millions of £.
- Numbers of computers and components are measured in thousands.

The decisions that the company has to make are represented by the following *decision variables*:

$$x_j = \text{number of thousands of laptops of type } j \text{ to be produced} \quad (j = 1, \ldots, 5).$$

Then the total profit (in millions of £) to be maximised is

$$2x_1 + 1.4x_2 + x_3 + 0.8x_4 + 0.5x_5.$$

Note that the quantities to be produced should be nonnegative, so the variables should satisfy the constraints

$$x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0,\ x_4 \geq 0,\ x_5 \geq 0.$$

The total number (in thousands of laptops) produced is

$$x_1 + x_2 + x_3 + x_4 + x_5$$

which has to be less than or equal to the number of CPUs available. That is

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq 8.$$

Similarly, the total number of thousands of SSDs used has to be less than or equal to the amount available. That is

$$x_1 + 0.7x_2 + 0.3x_4 \leq 3.$$

Proceeding similarly for the other constraints, we can write the problem that the company needs to solve.

$$\begin{array}{lrcl}
\text{maximise} & 2x_1+1.4x_2+x_3+0.8x_4+0.5x_5 & & \\
\text{subject to} & x_1+x_2+x_3+x_4+x_5 \leq 8 & & \text{(CPUs available)} \\
& x_1+0.7x_2+0.3x_4 \leq 3 & & \text{(SSDs available)} \\
& 6x_1+4x_2+4x_3+2x_4+2x_5 \leq 14 & & \text{(Memory boards availale)} \\
& x_1+x_2 \leq 5 & & \text{(Expected demand L1, L2)} \\
& x_3+x_4+x_5 \leq 4 & & \text{(Expected demand L3, L4, L5)} \\
& x_2 \geq 0.7 & & \text{(Orders for L2)} \\
& x_5 \geq 0.7 & & \text{(Orders for L5)} \\
& x_1,\ x_2,\ x_3,\ x_4,\ x_5 \geq 0. & &
\end{array}$$

Notice that we are not insisting that the company use all the components available. It is quite possible that it has more SSDs or memory boards than necessary, or that it is more profitable not to use all of a certain type of component.

Another aspect to consider is that computers can only be produced in integer quantities, whereas it may well be that the solution of the above problem indicates that we should produce fractional amounts of computers. For example, a value of 0.3334 for variable x_1 would indicate that we should produce 333.4 computers of type L1. However, in this case, if we just truncate the solution after the first three digits we will obtain a solution very close to the optimal one, so the discrete nature of the problem is not a major concern in this particular application.

In many other OR problems, however, the fact that certain variables should take an integer value is crucial (for example, if a variable represents a yes/no decision, and should take value either 0 or 1). These types of problems belong to the field of *integer programming*, and fall outside the scope of this course. The interested reader is referred to Chapter 11 in [HL].

Example 2 - Fixed income portfolio management

Solodrex has a budget of £1,000,000 that is considering investing into four bonds. The expected annual return, the worst-case annual return on each bond, and the "duration" of each bond are give in the table below. (In finance, the duration of a bond is a measure of the bond's sensitivity to interest rates, where a lower duration reflects a lower sensitivity and thus a lower risk.)

	Expected return	Worst case return	Duration
Bond 1	13%	6%	3
Bond 2	8%	8%	4
Bond 3	12%	10%	7
Bond 4	14%	9%	9

Solodrex wants to maximise the expected return from its bond investments, subject to three constraints:

1. The worst case return of the bond portfolio must be at least 8%.
2. The average duration of the portfolio must be at most 6.
3. Because of diversification requirements, at most 40% of the total amount invested can be invested in a single bond.

Constructing the formulation:

1. *What can the company decide to do?* How much to invest in each of the bonds.
2. *What is the company's objective?* To maximise expected return of the portfolio.
3. *What are the restrictions?* The budget available, the bound on the expected worst case return and duration, and the limit on how much can be invested on a single stock.

Decision variables: Let x_j, $j = 1, 2, 3, 4$, be the fraction of the total budget invested in Bond j. (So, $x_1 = 0.2$ means that we invest £200,000 in Bond 1.)

Constraints: We first need to impose that the total amount invested does not exceed our budget. This is expressed by the constraint

$$x_1 + x_2 + x_3 + x_4 \leq 1.$$

The total worst case return is $0.06x_1 + 0.08x_2 + 0.1x_3 + 0.09x_4$. This should be at least 0.08. Thus we need to impose the constraint

$$0.06x_1 + 0.08x_2 + 0.1x_3 + 0.09x_4 \geq 0.08.$$

Similarly, for the duration, we have the constraint $3x_1 + 4x_2 + 7x_3 + 9x_4 \leq 6$.

We need to impose that we do not invest in any bond more than 40% of the entire amount invested. Since the entire amount invested is $x_1 + x_2 + x_3 + x_4$, we need to impose that $x_j \leq 0.4(x_1 + x_2 + x_3 + x_4)$ for $j = 1, 2, 3, 4$. These four constraints can be expressed as

$$\begin{aligned} -0.6x_1 + 0.4x_2 + 0.4x_3 + 0.4x_4 &\geq 0 \\ 0.4x_1 - 0.6x_2 + 0.4x_3 + 0.4x_4 &\geq 0 \\ 0.4x_1 + 0.4x_2 - 0.6x_3 + 0.4x_4 &\geq 0 \\ 0.4x_1 + 0.4x_2 + 0.4x_3 - 0.6x_4 &\geq 0. \end{aligned}$$

Finally, all variables should be nonnegative.

Objective function: max $0.13x_1 + 0.08x_2 + 0.12x_3 + 0.14x_4$.

The formulation therefore is:

$$\begin{array}{rrrrrrrrrrr} \text{maximise} & 0.13x_1 & + & 0.08x_2 & + & 0.12x_3 & + & 0.14x_4 & & \\ \text{subject to} & x_1 & + & x_2 & + & x_3 & + & x_4 & \leq & 1 \\ & 0.06x_1 & + & 0.08x_2 & + & 0.1x_3 & + & 0.09x_4 & \geq & 0.08 \\ & 3x_1 & + & 4x_2 & + & 7x_3 & + & 9x_4 & \leq & 6 \\ & -0.6x_1 & + & 0.4x_2 & + & 0.4x_3 & + & 0.4x_4 & \geq & 0 \\ & 0.4x_1 & - & 0.6x_2 & + & 0.4x_3 & + & 0.4x_4 & \geq & 0 \\ & 0.4x_1 & + & 0.4x_2 & - & 0.6x_3 & + & 0.4x_4 & \geq & 0 \\ & 0.4x_1 & + & 0.4x_2 & + & 0.4x_3 & - & 0.6x_4 & \geq & 0 \\ & & & & & x_1,\ x_2,\ x_3,\ x_4 & & & \geq & 0. \end{array}$$

Alternate formulation: Observe that, in the above problem, it is never an optimal strategy to invest only a part of the total budget. This means that the constraint $x_1+x_2+x_3+x_4 \leq 1$ can instead be set at equality. The constraints expressing that we should not invest more than 40% in any bond j ($j = 1, 2, 3, 4$) can be now simply expressed by $x_j \leq 0.4$. We can therefore formulate the problem as follows:

$$\begin{array}{rrrrrrrrrrr} \text{maximise} & 0.13x_1 & + & 0.08x_2 & + & 0.12x_3 & + & 0.14x_4 & & \\ \text{subject to} & x_1 & + & x_2 & + & x_3 & + & x_4 & = & 1 \\ & 0.06x_1 & + & 0.08x_2 & + & 0.1x_3 & + & 0.09x_4 & \geq & 0.08 \\ & 3x_1 & + & 4x_2 & + & 7x_3 & + & 9x_4 & \leq & 6 \\ & & & & & x_1,\ x_2,\ x_3,\ x_4 & & & \leq & 0.4 \\ & & & & & x_1,\ x_2,\ x_3,\ x_4 & & & \geq & 0. \end{array}$$

Clearly, the two formulations are equivalent.

2.2 General form of a linear programming problem

A *linear programming (LP)* problem is to determine the values of the variables in order to maximise (or minimise) a linear objective function subject to linear constraints. Both examples given previously are linear programming problems.

Formally speaking, a general linear programming problem is of the form

$$\begin{array}{rrcl} \text{minimise or maximise} & c_1x_1 + \; c_2x_2 + \ldots + \; c_nx_n & & \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \gtrless & b_1 \\ & a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \gtrless & b_2 \\ & \vdots & & \\ & a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \gtrless & b_m \end{array}$$

where the number of variables is n, the number of constraints is m, any the symbol $\gtrless$ means any one of the $\geq$, $\leq$, or $=$ relations.

In matrix algebra terms, a linear programming problem is

$$\begin{array}{rl} \text{minimise or maximise} & cx \\ \text{subject to} & Ax \gtrless b \end{array}$$

where $c = (c_1, \ldots, c_n)$ is a n-dimensional row vector of the coefficients of the objective function, $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ is a n-dimensional column-vector of the variables, A is an $m \times n$ matrix, $\gtrless$ means a mixture of $\geq$, $\leq$, and $=$ constraints, and $b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$ is an m-dimensional column-vector of the right-hand-sides of the constraints.

2.2.1 Nonnegativity and resource constraints

We often distinguish the *nonnegativity constraints* (i.e. constraints of the form "variable ≥ 0") from the remaining constraints. The remaining constraints are referred to as *resource constraints*.

A problem in which all variables are nonnegative (which is often the case in practical applications) is of the form:

$$\begin{array}{rrcll} \text{minimise or maximise} & c_1x_1 + \; c_2x_2 + \ldots + \; c_nx_n & & & \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \gtrless & b_1 & \\ & a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \gtrless & b_2 & \\ & \vdots & & & \\ & a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \gtrless & b_m & \\ & x_j & \geq & 0, & j = 1, \ldots, n. \end{array}$$

where the number of variables is n, the number of resource constraints is m, and there are n nonnegativity constraints $x_j \geq 0$, $j = 1, \ldots, n$.

In matrix algebra terms, a linear programming problem were all variables are nonnegative can be written in the form

$$\begin{array}{rl} \text{minimise or maximise} & cx \\ \text{subject to} & Ax \gtrless b \\ & x \geq 0 \end{array}$$

where 0 denotes the n-dimensional column-vector of zeros.

A special type of resource constraint of the form "variable $\leq$ a given value" is referred to as an *upper-bound constraint*. For example, in the alternate formulation for the fixed income portfolio management problem, the constraints imposing that at most 40% of the budget is allocated on any bond, namely $x_j \leq 0.4$, $j = 1, 2, 3, 4$, are upper-bound constraints.

2.3 Assumptions of a linear programming model

1. *The variables represent quantities that are homogeneous and infinitely divisible.* This implies that variables can take any real value. For example, strictly speaking Example 1 does not satisfy this condition, because we can only produce computers in integer amounts, so the quantities are not infinitely divisible. So solving such problems as LP models is not efficient, nor guaranteed to give the best solution. This requires an integer programming model, as discussed previously.

2. *There are no economies or diseconomies of scale.* This means that returns or costs to scale are constant. In real world-applications, instead, it often the case that there are volume discounts when purchasing resources.

2.4 Alternative formulations

Formulating a problem as a linear program requires choosing appropriate decision variables. This choice, however, is not unique, and different choices of variables produce different formulations. We illustrate this in the following example.

Two processes simultaneously produce the products X and Y. The labour, machine-hours, electricity, and relative amounts of output produced by each process are given in the table below.

	Process I	Process II
Labour man-hours per 1000 X	10	15
Machine-hours per 1000 X	3	2
kilowatt-hours per 1000 X	25	18
Output of Y per 1000 X	500	750

The maximum amount of labour available is 150 man-hours, and of machine-hours is 30. There are no restrictions on the amount of electricity available or the amount of X and Y that can be sold. The costs and prices are given in the next table.

Wages per man-hour	£0.75
Price of electricity per kWh	£0.01
Selling price of X per thousand	£6.00
Selling price of Y per thousand	£9.00

Constructing the formulation:

1. *What is the owner's objective?* To maximise profit.
2. *What restricts his/her profit?* The amounts of labour and machine-hours.
3. *What can the owner decide to do?* The amounts of production from Process I and II.

It is necessary to decide on the units:

1. Profit is measured in £, as revenue minus cost.
2. Labour is measured in man-hours.
3. Machine time is measured in machine-hours.

Decision variables: Let x_1 represent the number of thousand X produced by process I, and let x_2 represent the number of thousand X produced by process II.
First we need to compute the net revenue of x_1 and x_2 (per 1000X).

Process I	Labour cost	$= 10 \cdot$ £0.75	= £7.50
	Electricity cost	$= 25 \cdot$ £0.01	= £0.25
	Total cost	=	£7.75
	Revenue from X	=	£6.00
	Revenue from Y	$= 0.5 \cdot$ £9.00	= £4.50
	Total revenue	=	£10.50
	Profit	=	£2.75

Process II	Labour cost	$= 15 \cdot$ £0.75	= £11.25
	Electricity cost	$= 18 \cdot$ £0.01	= £0.18
	Total cost	=	£11.43
	Revenue from X	=	£6.00
	Revenue from Y	$= 0.75 \cdot$ £9.00	= £6.75
	Total revenue	=	£12.75
	Profit	=	£1.32

The LP formulation is

$$\begin{array}{rl} \text{maximise} & cx \\ \text{subject to} & Ax \leq b \\ & x \geq 0 \end{array}$$

where $c = (2.75, 1.32)$, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $A = \begin{pmatrix} 10 & 15 \\ 3 & 2 \end{pmatrix}$, $b = \begin{pmatrix} 150 \\ 30 \end{pmatrix}$.

Alternative formulation: All that arithmetic can be saved by using different decision variables.

Decision variables:

p_1:	number of thousand X made by process I;
p_2:	number of thousand X made by process II;
s_1:	the number of thousands of X sold;
s_2:	the number of thousands of Y sold;
lab:	the number of man-hours purchased;
kw:	the number of kilowatt-hours purchased;
mc:	the number of machine-hours used.

All the variables are constrained to be nonnegative.

Constraints:

labour used:	$10p_1 + 15p_2 - lab \leq 0$
machine-hours used:	$3p_1 + 2p_2 - mc \leq 0$
electricity used:	$25p_1 + 18p_2 - kw \leq 0$
production X:	$p_1 + p_2 - s_1 \geq 0$
production Y:	$0.5p_1 + 0.75p_2 - s_2 \geq 0.$

The upper bounds are:

labour limit:	$lab \leq 150$
m/c limit:	$mc \leq 30.$

Objective function:

$$\text{maximise } 6s_1 + 9s_2 - 0.75lab - 0.01kw.$$

This second formulation has two major advantages. First, should a price change, there is only one coefficient to alter in the formulation, whereas in the first formulation, to incorporate a price change, you have to recalculate the coefficients. Second, it is user friendly - the output will be easier to interpret.

It is interesting to note that this formulation does not insist that all the labour purchased is used. There may be constraints elsewhere in the model on the amount of labour that has to be paid for. Similarly for the machine hours used. The constraint on electricity usage ensures that the amount used by the processes is the amount purchased. It could have been written as an equality because the positive cost of buying electricity would ensure that = will hold.

2.5 Exercises

Exercise 2.1. A manufacturer produces four types of cloth, A, B, C, and D, in various combinations of colours. One unit length of cloth uses hanks of coloured wool as follows:

Wool	Cloth Type				Number of hanks of wool available next week
	A	B	C	D	
Green	1	2	1	1	10
Red	2	1	2	1	6
Blue	3	1	0	0	10
Yellow	1	4	0	0	18
Brown	0	0	1	3	8
Purple	0	0	3	3	12

From one unit length of cloth the following profit is obtained:

From cloth	A	B	C	D
Profit (£)	3	5	4	1

Formulate the manufacturer's problem of determining which cloth to produce next week in order to maximise his profit.

Exercise 2.2. A manager has £1000 to invest, and he has a choice of three possible investments: A, B, and C. The expected value and the guaranteed minimum value after one year of a present investment of £100 are as follows:

	Expected	Guaranteed
A	£140	£90
B	£120	£120
C	£160	£40

The combined investment of B and C must be at least £600. The manager's aim is to maximise the expected value after one year, and also to achieve a guaranteed value of at least £1050. Formulate this problem as an LP.

Exercise 2.3. An oil firm has equipment for Process I with capacity of 2 tonnes per day, and for Process II with capacity of 3 tonnes per day. Process I requires 20 men to convert one tonne of crude oil into 3/4 tonne of A and 1/4 tonne of B, while Process II requires 10 men to convert one tonne of crude oil into 1/4 tonne of A and 3/4 tonne of B. The cost of these conversions is £60 per tonne by Process I and £30 per tonne by Process II.

There are 40 men and 4 tonnes of crude oil available per day. The selling price of A is £285 per tonne and of B is £105 per tonne. Formulate this problem as an LP.

Chapter 3

Feasibility, optimality, boundedness

In this lecture we discuss the possible outcomes of a linear programming problem, and introduce the concept of extreme points of an LP.

3.1 Feasible region

Consider the LP problem

$$\begin{array}{rrcl} \text{maximise} & 2x_1+8x_2 & & \\ \text{subject to} & 2x_1+\ x_2 & \leq & 10 \\ & x_1+2x_2 & \leq & 10 \\ & x_1+\ x_2 & \leq & 6 \\ & x_1+3x_2 & \leq & 12 \\ & 3x_1-\ x_2 & \geq & 0 \\ & x_1+4x_2 & \geq & 4 \\ & x_1, x_2 & \geq & 0. \end{array} \tag{3.1}$$

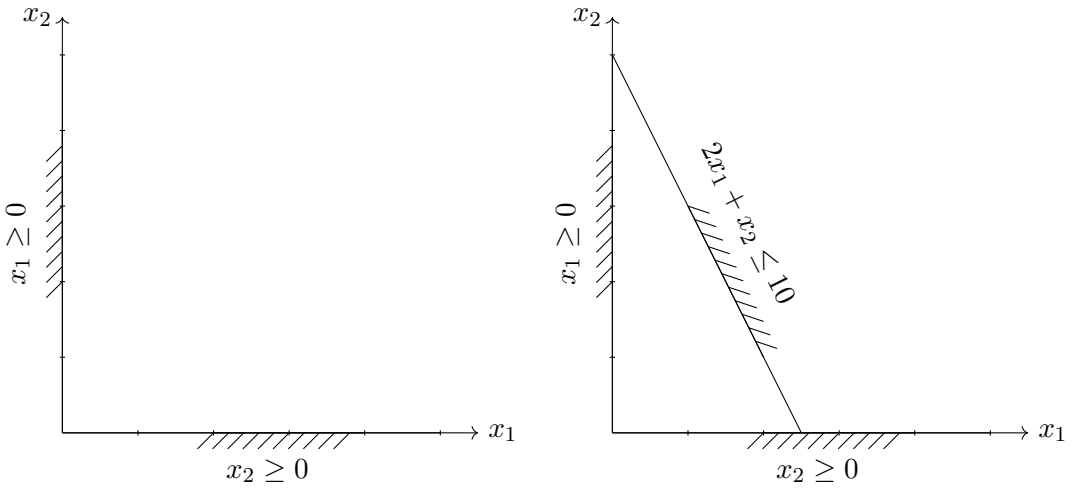
We can represent each constraint in (x_1, x_2) space. The nonnegativity constraints imply that only the positive quadrant including the axes needs to be considered. The hatching indicates the forbidden - infeasible - side of the line. See Figure 3.1(a).

A constraint $2x_1 + x_2 \leq 10$ can be represented by the line $2x_1 + x_2 = 10$ with hatching to indicate the disallowed side of the line, see Figure 3.1(b).

The 8 constraints (including the nonnegatives) restrict the allowed values of x_1 and x_2 to the feasible region lying on the un-hatched side of each line. That is the area bounded by the points ABCDEF in Figure 3.2.

Notice that the constraint $x_1 + 2x_2 \leq 10$ is redundant; if it were not there, the feasible region would be exactly the same. While this may be obvious in a diagram, it may be difficult to detect in problems with more than two variables. However, the availability of fast algorithms to solve Linear Programming makes it unnecessary to worry about removing redundant constraints.

Feasible points. A point x^* is feasible for a linear program if it satisfies all its constraints.



(a) The feasible region is contained in the nonnegative quadrant.

(b) Representation of one constraint.

Figure 3.1: Constraints defining the feasible region.

For example, the point $(1, 2)$ is feasible for the LP (3.1), as one can verify by substituting the values of x_1, x_2 and verifying that the inequalities are satisfied, as done below.

$$\begin{array}{rrcrc} 2 \cdot 1 + 2 & = 4 & < & 10 & \surd \\ 1 + 2 \cdot 2 & = 5 & < & 10 & \surd \\ 1 + 2 & = 3 & < & 6 & \surd \\ 1 + 3 \cdot 2 & = 7 & < & 12 & \surd \\ 3 \cdot 1 - 2 & = 1 & > & 0 & \surd \\ 1 + 4 \cdot 2 & = 9 & > & 4 & \surd \\ 1, 2 & & > & 0 & \surd \end{array}$$

On the other hand, the point $(\frac{72}{7}, -\frac{2}{7})$ is infeasible because it does not satisfy the nonnegativity constraint $x_2 \geq 0$ (one can verify that it satisfies all other constraints).

Feasible region. It is the set of all feasible points of a linear programming problem.

The feasible region of the LP (3.1) is depicted in gray in Figure 3.2.

3.2 The objective function as contours

In two dimensions, it is possible to represent the objective function by its contour lines. Consider that the objective function to be maximised is $2x_1+8x_2$. In Figure 3.3 we have drawn a set of sample contour lines. These are lines representing all points with some prescribed

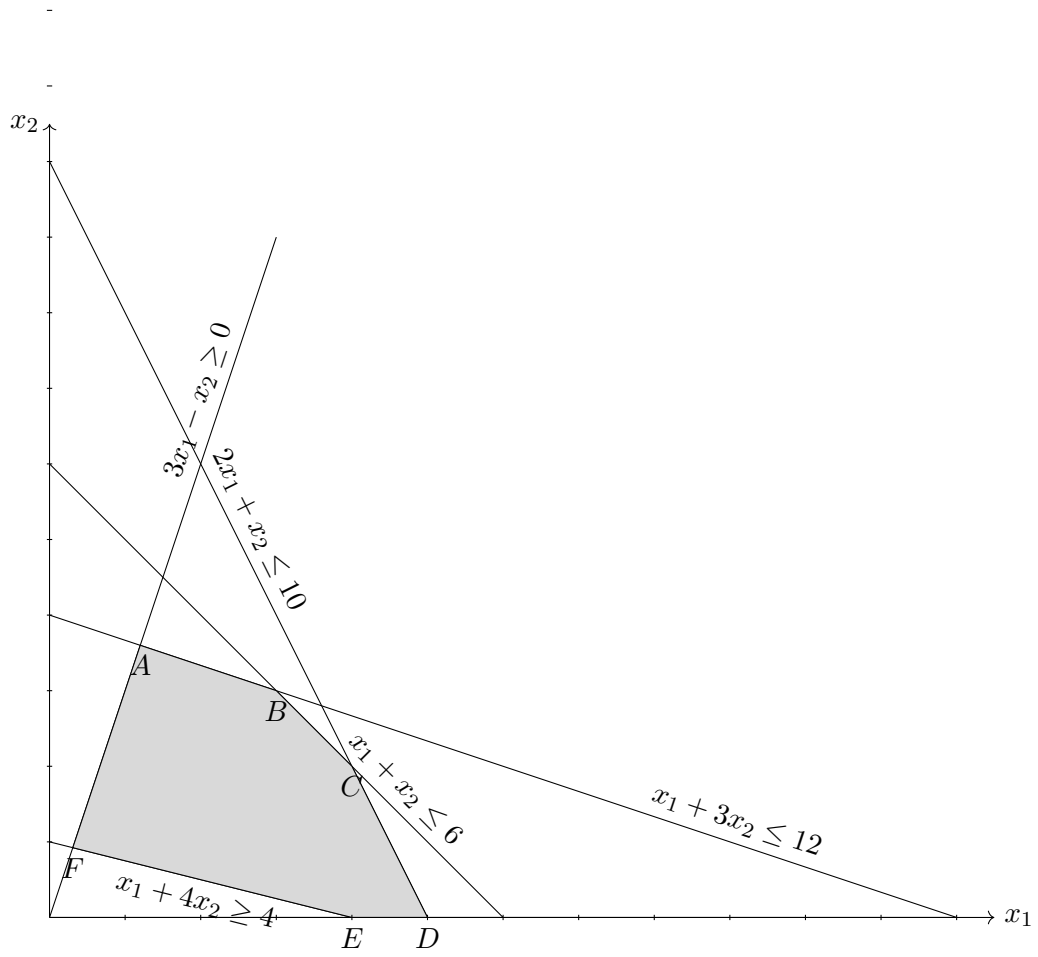


Figure 3.2: The feasible region is represented by the area shaded gray.

objective value, and they are of the form $2x_1 + 8x_2 = z$ for every choice of the parameter z representing the objective value.

There are an infinite number of contour lines that could be drawn. In the diagram, the arrow indicates the direction of increase of the objective function, that is the direction of increasing contours. Note that contour lines are parallel to each other, and they are perpendicular to the vector $(2, 8)$ of coefficients of the objective function.

3.3 The feasible region and the objective function

In Figure 3.4, the contours of the feasible region are superimposed onto the feasible region defined by the constraints of the LP (3.1). That is, the diagram in Figure 3.3 is superimposed on the diagram in Figure 3.2.

From the diagram in Figure 3.4, it appears that the optimum is point A, which is the intersection point of the lines corresponding to constraints 4 and 5. The values of x_1 and x_2

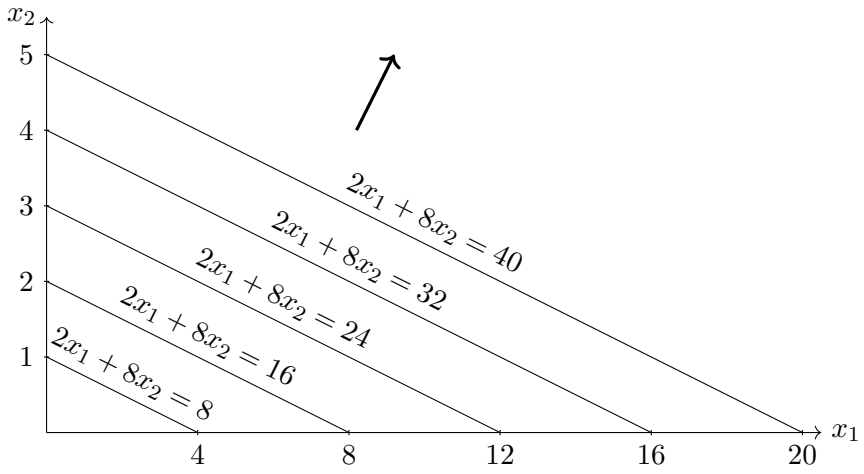


Figure 3.3: Contour lines of the objective function $2x_1 + 8x_2$. The arrow indicates the direction of maximisation.

at A are found by solving constraints 4 and 5 as equalities:

$$\begin{aligned} x_1 + 3x_2 &= 12 \\ -3x_1 + x_2 &= 0, \end{aligned}$$

giving $x_1 = 1.2$ and $x_2 = 3.6$. The value of the objective function at point $A = (1.2, 3.6)$ is therefore $2 \cdot 1.2 + 8 \cdot 3.6 = 31.2$.

Optimal solution. Given a maximisation LP problem, an *optimal solution* for the LP is a feasible point where the largest value of the objective function is taken among all feasible points.

Given a minimisation LP problem, an *optimal solution* for the LP is a feasible point where the smallest value of the objective function is taken among all feasible points.

The linear programming problem represented in Figure 3.4 has only one optimal solution, but observe that there might be cases were there are multiple optimal solutions. Consider the following example, where there are infinitely many optimal solutions.

$$\begin{aligned} \text{maximise} \quad & 2x_1 + 2x_2 \\ \text{subject to} \quad & x_1 + x_2 \leq 4 \\ & x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned} \tag{3.2}$$

Indeed, the optimal value for the problem is 8, and the optimal contour $2x_1 + 2x_2 = 8$ coincides with the border of the first constraint. Therefore all points in the line segment between the point $(1, 3)$ and the point $(4, 0)$ are optimal. See Figure 3.5(a).

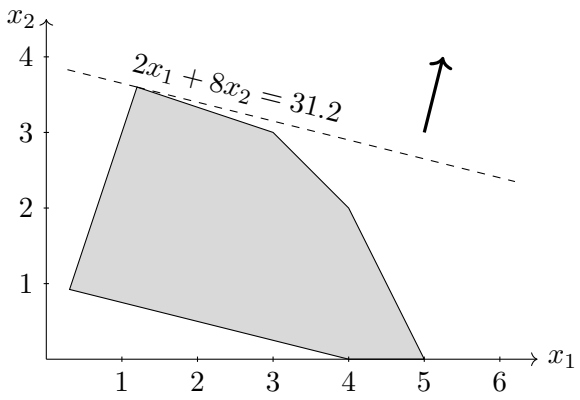


Figure 3.4: Feasible region, shaded in gray, and optimal contour of the objective function. The direction of maximisation is represented by the arrow perpendicular to the objective function contour.

3.4 Infeasible problems

It can happen that a linear programming problem has no feasible solution. In this case, we say that the problem is *infeasible*. Consider the following problem.

$$\begin{array}{rrcl} \text{maximise} & 2x_1 + x_2 & & \\ \text{subject to} & x_1 + x_2 & \leq & 4 \\ & x_2 & \geq & 5 \\ & x_1, x_2 & \geq & 0 \end{array} \tag{3.3}$$

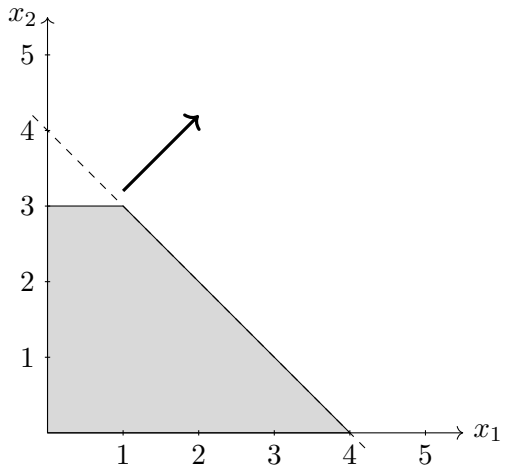
As can be easily seen from Figure 3.5(b), there is no point satisfying all four constraints, and therefore the problem is infeasible. Algebraically, one can see that the problem above is infeasible because if $x_2 \geq 5$ and x_1, x_2 are both nonnegative, then $x_1 + x_2$ must have value at least 5, and therefore the first constraint must be violated.

3.5 Unbounded problems

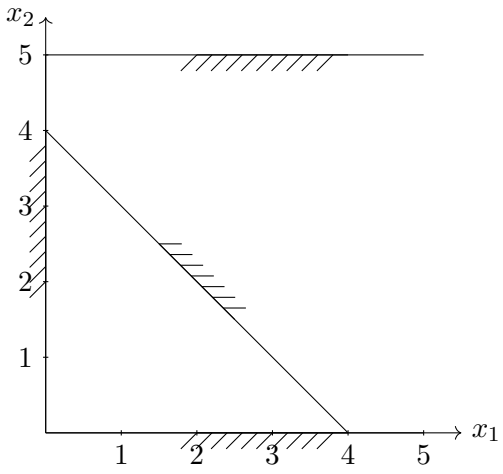
Consider the following LP problem.

$$\begin{array}{rrcl} \text{maximise} & x_1 + x_2 & & \\ \text{subject to} & x_1 + x_2 & \geq & 1 \\ & -x_1 + x_2 & \leq & 2 \\ & x_1 - 2x_2 & \leq & 2 \\ & x_1, x_2 & \geq & 0 \end{array} \tag{3.4}$$

The example is illustrated in Figure 3.6. From the diagram, it is evident that there exist feasible solutions for the LP with arbitrarily large value in the objective function.



(a) Problem (3.2) has infinitely many optima.



(b) Problem (3.3) is infeasible.

Figure 3.5: Illustrations of multiple optima and infeasibility.

Unbounded LP: A maximisation LP problem is *unbounded* if there exist feasible solutions with arbitrarily large value in the objective function.

A minimisation LP problem is *unbounded* if there exist feasible solutions with arbitrarily large negative value in the objective function.

Notice that, while the unboundedness of a linear program is a mathematical possibility, in real life, a message from the solver indicating that the problem is unbounded almost certainly indicates some mistake from the user in the mathematical formulation of the problem, typically the omission of one or more constraints.

3.6 Possible outcomes of a linear program

For any linear programming problem, exactly one of the following outcomes occurs:

- The problem has an optimal solution.
- The problem is infeasible.
- The problem is unbounded.

The above statement, true for linear programs, may be false for optimisation problems that are not linear. For example, consider the following (nonlinear) optimisation problem in

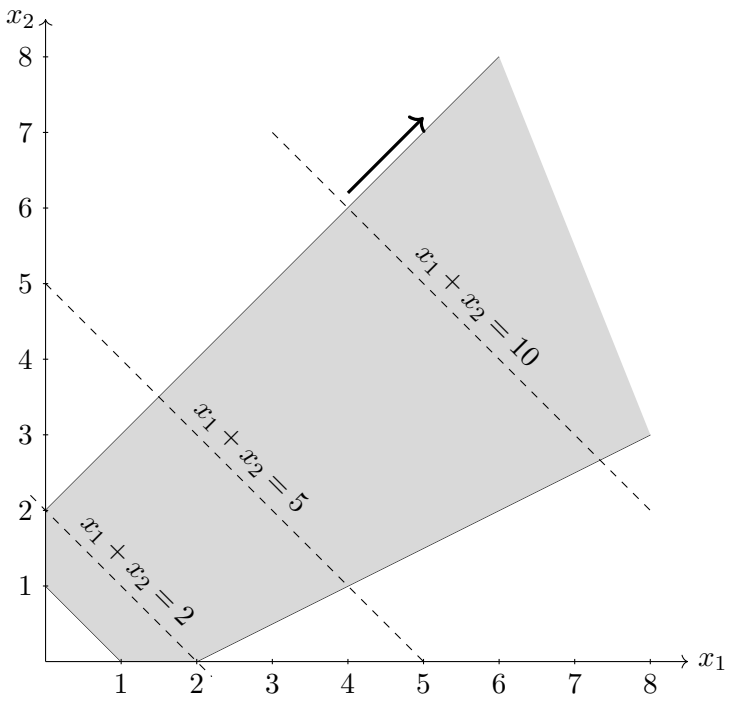


Figure 3.6: Problem (3.4) is unbounded.

the variables x_1, x_2.

$$\begin{array}{rc} \text{minimise} & x_2 \\ \text{subject to} & x_1 \cdot x_2 \geq 1 \\ & x_1, x_2 \geq 0 \end{array}$$

The feasible region is the area above the hyperbole of equation $x_1 x_2 = 1$, and it is represented in Figure 3.7. It can be observed that the problem is feasible (for example the point $x_1 = 1$, $x_2 = 1$ is a feasible solution), it is not unbounded, since it is a minimisation problem and no solution has value less than 0, but there is no optimal solution, because there are feasible solutions for which x_2 take values arbitrarily closed to 0 but no feasible solution with $x_2 = 0$ exists.

3.7 Basic solutions and extreme points

As we have seen in Section 3.3, the optimal solution of the LP (3.1) is the point A in Figure 3.2 of coordinates $x_1 = 1.2$, $x_2 = 3.6$. It can be observed from the diagram that this point is a corner point of the polygon that defines the feasible region. As we shall see, the fact that the optimum is achieved by some "corner point" of the feasible region is true in general,

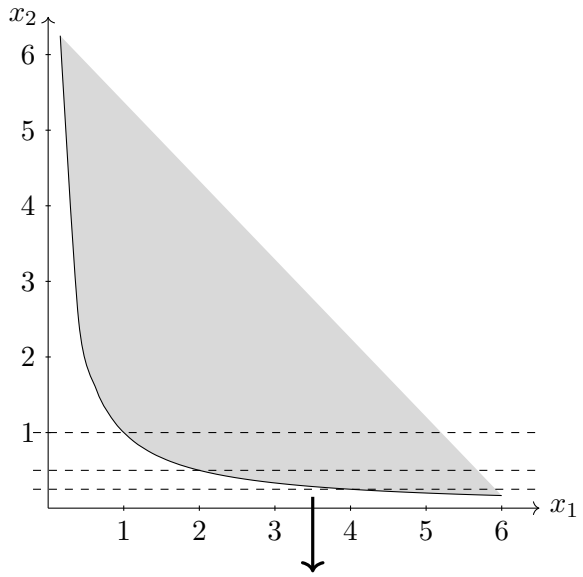


Figure 3.7: A nonlinear problem which is feasible, with bounded value, but has no optimal solution.

even for problems with more than two variables. Firstly, we need to define what we mean by "corner points", which in the language of Linear Programming are called *extreme points*.

In two dimensions, a corner point is one at the intersection of two non-parallel inequalities. In order to formalise the concept to the case of more variables, we need to recall the following concept from linear algebra.

Independent constraints. Given a system of n linear inequalities in n variables, we say that the n inequalities are *(linearly) independent* if there exists a unique solution satisfying all n of them at equality.

For example, the inequalities

$$\begin{array}{rrrcl} x_1 & +x_2 & & \leq & 2 \\ x_1 & & +x_3 & = & 2 \\ & x_2 & +x_3 & \geq & 2 \end{array}$$

are independent because the only solution satisfying all three of them at equality is the point $(x_1^*, x_2^*, x_3^*) = (1, 1, 1)$, as one can verify.

On the other hand, the inequalities

$$\begin{array}{rrrcl} x_1 & +x_2 & & \leq & 2 \\ x_1 & & +x_3 & = & 2 \\ & -x_2 & +x_3 & \geq & 0 \end{array}$$

are not independent because there are multiple solutions satisfying all three of them at equality. For example, the points $(1, 1, 1)$ and $(0, 2, 2)$.

Note that in two dimensions, two equations are independent if they do not define parallel lines. In the latter case, if the two parallel lines are distinct, then there are no solutions satisfying both equations; while if the two lines coincide, then there are infinitely many solutions satisfying both equations.

To simplify matters, in all our examples and in all exercises we will always work under the assumption that any n inequalities we deal with are independent.

Consider a system of linear constraints in n variables.

Basic point. A *basic point* for the system is a point satisfying at equality n independent constraints from the system.

Defining constraints. Given a basic point $\bar{x}$ for the system, a set of *defining constraints for* $\bar{x}$ is any choice of n independent constraints satisfied at equality by $\bar{x}$.

Example. Consider the LP in (3.1), whose feasible region is represented in Figure 3.2. The basic point labelled B is defined by the intersection of constraints 3 and 4, so it is the solution of the system:

$$\begin{aligned} x_1 + x_2 &= 6 \\ x_1 + 3x_2 &= 12. \end{aligned}$$

Basic point D is defined by constraint 1 and the nonnegativity of x_2, that is:

$$\begin{aligned} 2x_1 + x_2 &= 10 \\ x_2 &= 0. \end{aligned}$$

Note that, in the definition, we do not insist that a basic point is feasible for the whole system of constraints. For example, basic points B and D are feasible. Basic point G defined by

$$\begin{aligned} 2x_1 + x_2 &= 10 \\ x_1 + 3x_2 &= 12 \end{aligned}$$

is infeasible.

Extreme point. A point that is both basic and feasible for a given system of linear constrains.

For example, points A to F in Figure 3.2 are extreme points while points G and the origin are basic, but they are not extreme.

3.7.1 Is a point an extreme point?

For a system of linear constraints in n-variables $x_1, \ldots, x_n$, to determine whether a point x^* is an extreme point:

- Substitute the values of $x_1, x_2, \ldots, x_n$ into the constraints, and verify if the point is feasible.
- Determine whether n (or more) of them are satisfied as equalities.
- Among the constraints satisfied at equality by x^*, determine if there are n of them that are independent.

For example, consider as usual the system of constraints (3.1), and let $x^* = (4, 2)$. Substituting the values $x_1 = 4$, $x_2 = 2$ into the constraints, we obtain:

$$\begin{array}{rlcll} 2 \cdot 4 + 2 & = 10 & = & 10 & \surd \\ 4 + 2 \cdot 2 & = 8 & \leq & 10 & \\ 4 + 2 & = 6 & = & 6 & \surd \\ 4 + 3 \cdot 2 & = 10 & \leq & 12 & \\ 3 \cdot 4 - 2 & = 10 & \geq & 0 & \\ 4 + 4 \cdot 2 & = 12 & \geq & 4 & \\ 4, 2 & & \geq & 0. & \end{array}$$

The point is basic, as constraint 1 and constraint 3 are satisfied as equalities and they are independent. The point (4, 2) is point C in the diagram in Figure 3.2. Point C is an extreme point, as these calculations show that it is feasible as well as basic.

Note. In general, a basic point might have more than one set of defining constraints. For example, point $\bar{x} = (1, 1, 1)$ is basic for the following system:

$$\begin{array}{rrrcl} x_1 & +x_2 & & \leq & 2 \\ x_1 & & +x_3 & = & 2 \\ & x_2 & +x_3 & \geq & 2 \\ x_1 & -x_2 & +x_3 & \geq & 1, \end{array}$$

but each subset consisting of three constraints is defining for $\bar{x}$, since $\bar{x}$ satisfies at equality all four constraints and every three of them are independent.

3.7.2 Optimality and extreme points

The following is one of the most fundamental and useful facts in Linear Programming.

Extreme optimal solutions. Whenever an LP admits an optimal solution, there exists some optimal solution that is an extreme point of the feasible region.

This fact tells us that, when solving an LP, one only needs to search among the extreme points, and pick the one with best objective value.

Note. A linear programming problem may have non-extreme points that are also optimal, but according to the statement above, there is always at least an extreme point on the optimal contour. For example, the LP in (3.2) has two extreme points that are optimal, namely the points $(1, 3)$ and $(4, 0)$, but there are also infinitely many optimal solution that are not extreme, namely all points in the line segment between $(1, 3)$ and $(4, 0)$ (see Figure 3.8).

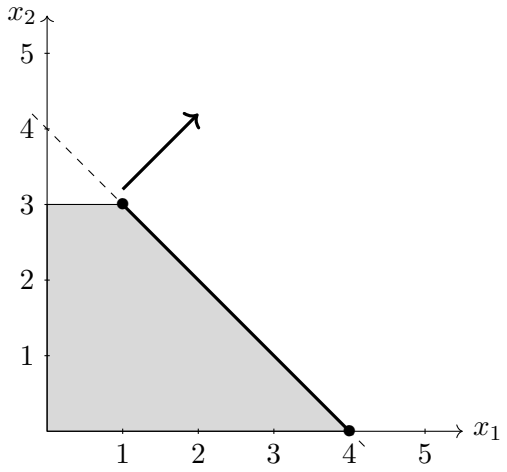


Figure 3.8: Problem (3.2) has two extreme optimal solutions and infinitely many non-extreme optimal solutions.

3.8 Exercises

Exercise 3.1. A firm produces commodities A and B from the same raw material. It has two plants available which produce, simultaneously, the following amounts per hour:

	A	B
Plant 1	5	2
Plant 2	3	6

The cost of running Plant 1 is £20 per hour in normal time, and £30 per hour in overtime. For Plant 2, the cost is £40 per hour in normal time, and £60 per hour overtime. There are 8 hours of normal time per day. 30 units of A and 60 units of B per day are required.

Formulate this problem as a LP. Why does your formulation ensure that in each plant normal time will be used in preference to overtime?

Exercise 3.2. Draw the feasible region in two dimensional space bounded by the following constraints.

$$\begin{array}{rrcl} x_1+ & x_2 & \leq & 6 \\ x_1- & 5x_2 & \leq & 0 \\ 2x_1+ & x_2 & \geq & 6 \\ -3x_1+ & 4x_2 & \leq & 3 \\ & x_1, x_2 & \geq & 0 \end{array}$$

Are the following points feasible? Extreme points? In case of extreme points, which constraints define them?

(i) $x_1 = 3, x_2 = 3$.

(ii) $x_1 = 1, x_2 = 3$.

(iii) $x_1 = 0, x_2 = 0$.

(iv) $x_1 = 5, x_2 = 1$.

What is the feasible region if the additional constraint $x_2 \geq 4$ is added to the system?

Exercise 3.3. A feasible region in (x_1, x_2) space is bounded only by the two constraints $x_1 + x_2 \leq 5$ and $x_2 \geq 2$. These define only one extreme point: $(3, 2)$. For what values of a and b will the linear function $ax_1 + bx_2$ be maximised at this point?

Chapter 4

Optimality

So far, we have been able to establish optimality by graphical means. This is clearly unsatisfactory since the method only works for problems with two variables, and even for two variables there are cases were our diagram might not be very accurate. We have seen in the previous lecture that, in order to solve an LP, it is sufficient to search among the extreme points, and pick the one with best value. Since the number of extreme points is finite, this gives a solution method, provided that we are able to answer the following question: given an extreme point of an LP, is the point optimal? Answering this question will be the main focus of this lecture.

4.1 Proving optimality

Consider again the LP (3.1), which is represented in Figure 4.1.

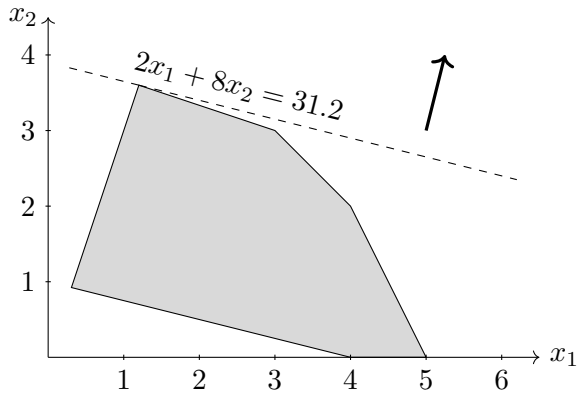


Figure 4.1: The feasible region lies in the area defined by $2x_1 + 8x_2 \leq 31.2$.

From the diagram, we can infer that the optimal solution has value 31.2. Since (3.1) is

a maximisation problem, this means that every feasible solution has value less than or equal to 31.2. In other words, all the feasible region lies within the area defined by

$$2x_1 + 8x_2 \leq 31.2,$$

and there exists at least one point on the boundary.

In general, for a maximisation LP with objective function $c_1x_1 + c_2x_2 + ... + c_nx_n$ and optimal value γ^*, the feasible region lies within the area defined by the inequality

$$c_1x_1 + c_2x_2 + ... + c_nx_n \leq \gamma^*,$$

and the optimal solutions lie on the boundary.

For minimisation problems with objective function $c_1x_1 + c_2x_2 + ... + c_nx_n$ and optimal value γ^*, the feasible region lies within the area defined by the inequality

$$c_1x_1 + c_2x_2 + ... + c_nx_n \geq \gamma^*,$$

and the optimal solutions lie on the boundary.

Note that when maximising, the feasible region is on the $\leq$ side of the optimal contour and when minimising, the feasible region is on the $\geq$ side of the optimum contour. So, if we are given an extreme point x^* and we want to decide if it is optimal for a maximisation problem, we need a tool for deciding if the feasible region lies on the $\leq$ side of the objective function's contour passing through x^*. Such a tool is illustrated next.

4.1.1 Derived inequalities

Combining uniform inequalities. Consider a system of linear constraints, comprising constraints of type $\geq$, $\leq$, or $=$. One can add together positive and negative multiples of equality constraints, and derive a new equation that is satisfied by the same points that satisfy the original constraints. It is not possible to do exactly the same with inequalities. However, if we have a uniform set of linear inequalities, for example all expressed as $\leq$, then we can add together nonnegative multiples of them to derive a new $\leq$ inequality that is satisfied by the same points that satisfy the original constraints.

For example, consider the two inequalities

$$\begin{aligned} 2x_1 + x_2 &\leq 10 \\ x_1 + 3x_2 &\leq 12. \end{aligned}$$

The corresponding lines intersect at $x_1 = 3.6$, $x_2 = 2.8$. Deriving a new constraint with arbitrarily chosen weights of 2 for the first constraint and 1 for the second gives

$$\begin{array}{rrcl} 2 \cdot (& 2x_1 + x_2 & \leq & 10) \\ 1 \cdot (& x_1 + 3x_2 & \leq & 12) \\ \hline & 5x_1 + 5x_2 & \leq & 32. \end{array}$$

The point $x_1 = 3.6$, $x_2 = 2.8$ satisfies $5x_1 + 5x_2 \leq 32$ as an equality. This relationship is illustrated in Figure 4.2.

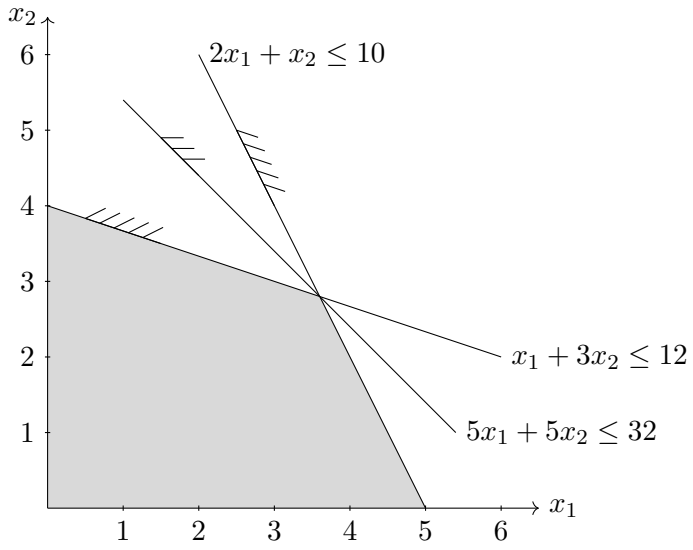


Figure 4.2: Deriving new constraints

We now have the ingredients for a test to determine whether or not a given point is optimal. Ask the question: is it possible to represent the optimal contour of the objective function as the nonnegative combination of the constraints that are satisfied at equality by the point? If the answer is yes, then the point is optimal, if the answer is no then the point is not optimal.

Note that in general an LP does not have all inequalities of the form $\leq$. However, we can turn the problem into such a form by applying the two following "tricks".

Multiplying a constraint by -1. The feasible area defined by a constraint is unaltered if it is multiplied throughout by -1. Notice that the direction of the inequality will change. That is, the area defined by the inequality

$$3x_1 + x_2 \geq 6$$

is identical to that defined by

$$-3x_1 - x_2 \leq -6.$$

Similarly, the nonnegativity $x_1 \geq 0$ is identical to $-x_1 \leq 0$. In general, the area defined by the linear inequality

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

is identical to that defined by

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

Equality constraints. An equality constraint can be expressed as two inequalities. For example the single constraint:

$$2x_1 + 7x_2 = 23$$

can be expressed as the two inequalities:

$$\begin{aligned} 2x_1 + 7x_2 &\leq 23 \\ 2x_1 + 7x_2 &\geq 23. \end{aligned}$$

These two inequalities can be expressed as two less than or equal constraints:

$$\begin{aligned} 2x_1 + 7x_2 &\leq 23 \\ -2x_1 - 7x_2 &\leq -23. \end{aligned}$$

In general, the equality

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

can be expressed as two less than or equal constraints:

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

4.1.2 Example of optimality proof

We consider again the problem (3.1), represented in Figure 4.1. We will give an algebraic proof that the optimal solution is point A in the diagram in Figure 3.2. We first write all constraints so that they are expressed as $\leq$ constraints, that is,

$$\begin{array}{lrcl} \text{maximise} & 2x_1 + 8x_2 & & \\ \text{subject to} & 2x_1 + x_2 & \leq & 10 \\ & x_1 + 2x_2 & \leq & 10 \\ & x_1 + x_2 & \leq & 6 \\ & x_1 + 3x_2 & \leq & 12 \\ & -3x_1 + x_2 & \leq & 0 \\ & -x_1 - 4x_2 & \leq & -4 \\ & -x_1 + 0x_2 & \leq & 0 \\ & -0x_1 - x_2 & \leq & 0. \end{array} \tag{4.1}$$

Select candidate extreme point: From the diagram in Figure 3.4, it appears that the optimum is point A, which is the extreme point defined by constraints 4 and 5. The values of x_1 and x_2 at A are found by solving constraints 4 and 5 as equalities

$$\begin{aligned} x_1 + 3x_2 &= 12 \\ -3x_1 + x_2 &= 0 \end{aligned}$$

giving $x_1 = 1.2$ and $x_2 = 3.6$. The value of the objective function at point A is therefore $2 \cdot 1.2 + 8 \cdot 3.6 = 31.2$.

Check feasibility: Substituting the values $x_1 = 1.2$ and $x_2 = 3.6$ into the constraints of the problem, we can verify that A is indeed feasible. Therefore A is an extreme point, and hence a candidate for an optimal solution.

Check optimality: In order to prove optimality of point A, we need to show that all feasible solutions satisfy the inequality

$$2x_1 + 8x_2 \leq 31.2.$$

Can we find nonnegative weights of constraints 4 and 5 (which define point A), so that the weighted combination of the two constraints produces the above inequality? We require to find nonnegative numbers y_4 and y_5 such that:

$$\begin{array}{rcl} y_4 \cdot (\ x_1 + 3x_2 & \leq & 12) \\ y_5 \cdot (-3x_1 + x_2 & \leq & 0) \\ \hline 2x_1 + 8x_2 & \leq & 31.2. \end{array}$$

The coefficient of x_1 in the inequality derived using y_4 and y_5 is $1 \cdot y_4 + (-3) \cdot y_5$, and this must equal 2. The coefficient of x_2 is $3 \cdot y_4 + 1 \cdot y_5$, and it must equal 8. It follows that, in order to find y_4 and y_5, we must solve the equations

$$\begin{array}{rcl} y_4 - 3y_5 & = & 2 \\ 3y_4 + y_5 & = & 8 \end{array}$$

giving $y_4 = 2.6$, $y_5 = 0.2$. Note that the right-hand-side of the inequality derived using $y_4 = 2.6$, $y_5 = 0.2$ is $12y_4 + 0y_5 = 31.2$.

Notice that we formed these equations by reading down the columns. Since the values of y_4 and y_5 that we determined are both nonnegative, we have succeeded in finding weights which enable us to represent the objective function as a derived constraint. So every point that is feasible with respect to the 2 constraints:

$$\begin{array}{rcl} x_1 + 3x_2 & \leq & 12 \\ -3x_1 + x_2 & \leq & 0 \end{array}$$

lies on the feasible side of $2x_1 + 8x_2 \leq 31.2$. As we have seen, the point $A = (1.2, 3.6)$ satisfies the inequality $2x_1 + 8x_2 \leq 31.2$ at equality, and we can therefore conclude that the objective function is maximised over the feasible region at the point A, with value 31.2.

4.1.3 Checking optimality: general statement

Consider a generic maximisation LP problem with n variables $x_1, \ldots, x_n$ and m constraints, of the form:

$$\begin{array}{lrcl} \text{maximise} & c_1x_1 + c_2x_2 + \ldots + c_nx_n & & \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \leq & b_1 \\ & a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \leq & b_2 \\ & & \vdots & \\ & a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \leq & b_m. \end{array} \tag{4.2}$$

(In particular, we are assuming that the nonnegatives are part of the constraints, and are written in the form $-x_j \leq 0$.)

Suppose we are given an extreme point x^* as a proposed optimal solution. Let γ^* be the value of the objective function at x^*, i.e. $\gamma^* = c_1x_1^* + \ldots + c_nx_n^*$. Select n constraints defining x^* among the m constraints of the LP (4.2), say the n constraints indexed $i_1, i_2, \ldots, i_n$. Thus, in the example from the previous section, $n = 2$, $i_1 = 4$, and $i_2 = 5$.

In order to verify that x^* is optimal, we need to combine these n inequalities using nonnegative multipliers, in order to obtain a new inequality defining the optimal contour. That is, we need to find nonnegative multipliers $y_{i_1}, y_{i_2} \ldots, y_{i_n}$ such that:

$$\begin{array}{rrrrrrr}
y_{i_1}\cdot & (a_{i_1 1}x_1 + & a_{i_1 2}x_2 + & \ldots & +a_{i_1 n}x_n & \leq & b_{i_1}) \\
y_{i_2}\cdot & (a_{i_2 1}x_1 + & a_{i_2 2}x_2 + & \ldots & +a_{i_2 n}x_n & \leq & b_{i_2}) \\
\vdots & & & & & & \\
y_{i_n}\cdot & (a_{i_n 1}x_1 + & a_{i_n 2}x_2 + & \ldots & +a_{i_n n}x_n & \leq & b_{i_n}) \\
\hline
= & (\quad c_1x_1 + & \quad c_2x_2 + & \ldots & +c_nx_n & \leq & \gamma^*).
\end{array}$$

To ensure that the inequality derived using multipliers $y_{i_1}, y_{i_2} \ldots, y_{i_n}$ has right-hand-side equal to $c_1x_1 + c_2x_2 + \ldots c_nx_n$, we must solve the system of n equations:

$$\begin{array}{rcl}
a_{i_1 1}y_{i_1} + a_{i_2 1}y_{i_2} + \ldots + a_{i_n 1}y_{i_n} & = & c_1 \\
a_{i_1 2}y_{i_1} + a_{i_2 2}y_{i_2} + \ldots + a_{i_n 2}y_{i_n} & = & c_2 \\
 & \vdots & \\
a_{i_1 n}y_{i_1} + a_{i_2 n}y_{i_2} + \ldots + a_{i_n n}y_{i_n} & = & c_n
\end{array}$$

to give the solution $y_{i_1}^*, \ldots, y_{i_n}^*$. Note that, since x^* satisfies at equality all the constraints $i_1, \ldots, i_n$, and since $c_1x_1^* + \cdots + c_nx_n^* = \gamma^*$, it follows automatically that $b_{i_1}y_1^* + \cdots + b_{i_n}y_n^* = \gamma^*$. There are two possible outcomes.

a) If $y_{i_1}^*, \ldots, y_{i_n}^* \geq 0$, then we have proved optimality of x^*. Indeed, by combining the constraints $i_1, \ldots, i_n$ using the nonnegative weights $y_{i_1}^*, \ldots, y_{i_n}^*$, we have derived the inequality

$$c_1x_1 + \cdots + c_nx_n \leq \gamma^*,$$

which shows that no feasible solution can have objective value greater than γ^*. Since the point x^* has indeed value γ^*, it follows that the function is maximised at the point x^* and the optimal value of the objective function is γ^*.

b) If some of the values $y_{i_1}^*, \ldots, y_{i_n}^*$ are negative, then we need to try with a different choice of n constraints defining x^* and repeat. If we were not able to prove optimality after trying all possible choices of n constraints among the ones satisfied at equality by x^*, then we conclude that x^* cannot be an optimal solution.

4.1.4 A demonstration of non-optimality

Point C in Figure 3.2 is the intersection of the lines corresponding to constraints 1 and 3:

$$\begin{array}{rcl}
2x_1 + x_2 & = & 10 \\
x_1 + x_2 & = & 6
\end{array}$$

which, when solved, give $x_1 = 4$ and $x_2 = 2$. Basic point C is feasible, and hence it is an extreme point and a candidate optimal solution. Note that the value of the objective function at point C is $2 \cdot 4 + 8 \cdot 2 = 24$. Is the function maximised at C? In order to answer this question, we need to find multipliers y_1, y_3 such that:

$$\begin{array}{rrcl} y_1 \cdot (& 2x_1 + x_2 & \leq & 10) \\ y_3 \cdot (& x_1 + x_2 & \leq & 6) \\ \hline & 2x_1 + 8x_2 & \leq & 24. \end{array}$$

We need to solve the following system of equations in the variables y_1, y_3:

$$\begin{array}{rcl} 2y_1 + y_3 & = & 2 \\ y_1 + y_3 & = & 8 \end{array}$$

which, when solved, gives $y_1 = -6$ and $y_3 = 14$. The negative value for y_1 shows that it is not possible to represent the objective function as a nonnegative combination of constraints 1 and 3. Observe that constraints 1 and 3 are the only ones satisfied at equality by point C in the LP (4.1), therefore there is no other possible choice of two independent constraints defining C. We conclude that the function is not maximised at the point C.

4.1.5 A degenerate example

Consider the LP:

$$\begin{array}{lrrcl} \text{maximise} & x_1 - & x_2 & & \\ \text{subject to} & x_1 + & x_2 & \leq & 1 \\ & x_1 & & \leq & 1 \\ & & -x_2 & \leq & 0. \end{array} \tag{4.3}$$

The problem is represented in the Figure 4.3.

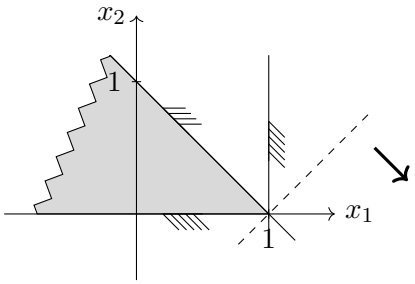


Figure 4.3: Representation of the LP (4.3). The jagged contour of the shaded area indicates that the feasible region continues indefinitely.

It is immediately clear from the figure that the optimal solution is the point $x^* = (1, 0)$, with objective value equal to 1. This problem is "degenerate" in the sense that x^* satisfies at equality three constraints, whereas the problem has only two variables. Therefore there

are three choices of defining constraints for x^*, namely: constraints 1 and 2; constraints 1 and 3; constraints 2 and 3. However, note that not every choice of defining constraints for x^* provides an optimality proof. Indeed, consider the three cases.

- Constraints 1 and 2: in this case, to prove optimality we need to find multipliers y_1, y_2 such that:

$$\begin{array}{rcl} y_1 \cdot (x_1 + x_2 & \leq & 1) \\ y_2 \cdot (\ x_1 & \leq & 1) \\ \hline x_1 - x_2 & \leq & 1. \end{array}$$

  Clearly, the only solution is $y_1 = 2$, $y_2 = -1$. Since y_2 is negative, this choice of defining constraints does not prove optimality of x^*.

- Constraints 1 and 3: in this case, to prove optimality we need to find multipliers y_1, y_3 such that:

$$\begin{array}{rcl} y_1 \cdot (x_1 + x_2 & \leq & 1) \\ y_3 \cdot (\quad -x_2 & \leq & 0) \\ \hline x_1 - x_2 & \leq & 1. \end{array}$$

  Clearly, the only solution is $y_1 = 1$, $y_3 = 2$. Since y_1 and y_3 are nonnegative, this choice of defining constraints proves optimality of x^*.

- Constraints 2 and 3: in this case, to prove optimality we need to find multipliers y_2, y_3 such that:

$$\begin{array}{rcl} y_2 \cdot (\ x_1 & \leq & 1) \\ y_3 \cdot (\quad -x_2 & \leq & 0) \\ \hline x_1 - x_2 & \leq & 1. \end{array}$$

  Clearly, the only solution is $y_2 = 1$, $y_3 = 1$. Since y_2 and y_3 are nonnegative, this choice of defining constraints proves optimality of x^*.

Figure 4.4 illustrates the three LPs obtained by only considering each of the three possible choices of the defining constraints. As can be seen from the figure, for the LP comprised of only constraints 1 and 2, the point $x^* = (0, 1)$ is not optimal, which explains why this choice of defining constraints does not prove optimality. For the LP comprised of only constraints 1 and 3, instead, the point $x^* = (1, 0)$ is optimal, and similarly for the LP comprised of only constraints 2 and 3.

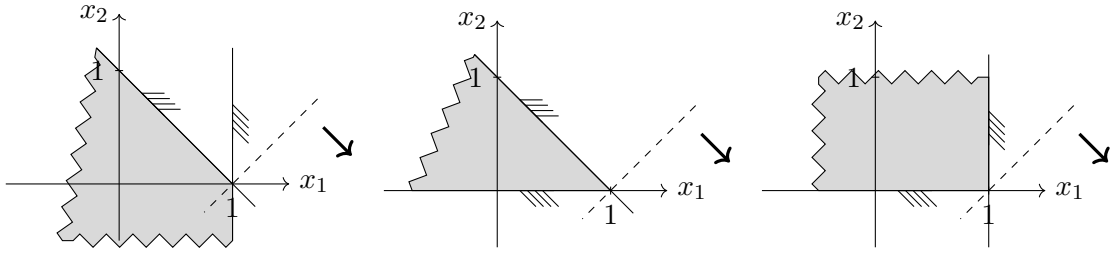


Figure 4.4: From left to right, the above diagrams represent the LPs comprised only of, respectively, constraints 1 and 2, constraints 1 and 3, constraints 2 and 3.

4.2 Solving maximisation linear programs

The previous discussion suggests the following method to solve linear programs.

Solving a maximisation LP in n variables

1. Convert all the m resource constraints, and the n nonnegatives, to $\leq$ constraints.
2. Identify n constraints defining a basic point which is a candidate for an optimal solution. Determine the values of the variables, $x_1^*, \ldots, x_n^*$, by solving the n inequalities as equalities.
3. Check that the point x^* is feasible. If the point is infeasible, go to step 2.
4. Check if x^* is optimal, as explained in Section 4.1.3.
5. If x^* is optimal, then STOP. Otherwise return to step 2 and try a new combination of n constraints.

- Observe that, for problems with two variables, one can first draw a diagram representing the LP in order to select the candidate optimal solution and the two constraints defining it.
- For problems with more than two variables, it is generally difficult to select "a priori" a good candidate optimal solution. In this case, one would have to try all possible combinations of n constraints until one finds a combination that defines an optimal extreme point. While this guarantees that an optimal solution will eventually be found, the number of possible combinations to consider will blow up exponentially as the number of variables increases.
- Linear programming solvers, while essentially carrying out the steps described above, have a clever way of enumerating the possible combinations of n constraints, known as the *Simplex method*. The Simplex method greatly reduces the number of arithmetic computations needed at every iteration of our general procedure, and in practice it ensures that only few possible candidate extreme points need to be considered before finding an optimal solution.

4.2.1 $\geq$ and $=$ constraints

In the solution method previously described, we require that all constraints are first converted to $\leq$. This is not necessary, but one needs to be careful about the sign of the weights assigned to the constraints.

$\geq$ constraints. In order to prove optimality of the extreme point defined by constraints 4 and 5 in problem (3.1), we first had to transform the problem so that all constraints were of

the form $\leq$. Therefore, constraint

$$3x_1 - x_2 \geq 0$$

of problem (3.1) was replaced by

$$-3x_1 + x_2 \leq 0.$$

That is, we multiplied the coefficients by -1, and reversed the sign of the inequality. In Section 4.1.2, we found nonnegative weights by which to multiply two constraints to derive the objective function. These were:

$$\begin{array}{rcl} 2.6 \cdot (\quad x_1 + 3x_2 & \leq & 12) \\ 0.2 \cdot (-3x_1 + x_2 & \leq & 0) \\ \hline 2x_1 + 8x_2 & \leq & \gamma. \end{array}$$

Now,

$$0.2 \cdot (-3x_1 + x_2 \leq 0)$$

is the same as

$$-0.2 \cdot (3x_1 - x_2 \geq 0).$$

This example shows that if the constraint is a $\geq$ constraint, we need a nonpositive weight, that is $y_i \leq 0$, to prove that the point is the optimum.

= constraints. The area defined by

$$x_1 + x_2 = 6$$

is the same area defined by

$$-x_1 - x_2 = -6$$

which is obtained by multiplying by -1 and making no changes to the sign of the inequality. Thus, if the optimum point is defined by an equality constraint, to prove optimality the weigh y_i of the constraint can be of either sign. The weight of an equality constraint is *free*, it can be any value: negative, zero, or positive.

4.2.2 Dual values

When proving optimality of an extreme point, we select a set of defining constraints for the point and assign multipliers to each defining constraint. When we discuss linear programming duality (Chapter 6), it will be convenient to assume that all inequalities are assigned a weight, so we consider the weight of the inequalities that do not define the extreme point to be 0.

For example, in the proof of optimality of the point $x_1 = 1.2$, $x_2 = 3.6$ for problem (3.1), we can assign dual values to all constraints, not just constraints 4 and 5, as follows:

$$
\begin{array}{rcl}
0 \cdot (2x_1 + x_2 & \leq & 10) \\
0 \cdot (x_1 + 2x_2 & \leq & 10) \\
0 \cdot (x_1 + x_2 & \leq & 6) \\
2.6 \cdot (x_1 + 3x_2 & \leq & 12) \\
-0.2 \cdot (3x_1 - x_2 & \geq & 0) \\
0 \cdot (x_1 + 4x_2 & \geq & 4) \\
0 \cdot (x_1 + 0x_2 & \geq & 0) \\
0 \cdot (0x_1 + x_2 & \geq & 0) \\
\hline
2x_1 + 8x_2 & \leq & 31.2.
\end{array}
$$

From now onwards the terms *weights, y variables*, and *dual variables* will be used interchangeably. They refer to the same objects.

In Table 4.1, we summarise the sign that the dual variables must have in order to prove optimality of a feasible extreme solution in a maximisation LP.

Constraint	**Dual variable**
$\leq$	nonnegative
$\geq$	nonpositive
$=$	free
Not defining the extreme point	0

Table 4.1: Summary of the dual variables when maximising.

4.3 Exercises

Exercise 4.1. Consider the following LP:

$$
\begin{array}{lrcl}
\text{maximise} & x_2 & & \\
\text{subject to} & x_1 + 2x_2 & \leq & 5 \\
& x_1 + x_2 & \leq & 3 \\
& -x_1 + x_2 & \leq & 1 \\
& x_1, x_2 & \geq & 0.
\end{array}
$$

Find the optimal solution by drawing the feasible region and the contours of the function to be maximised, then *prove* the solution to be feasible and optimal. If the proposed optimal solution is at the intersection of more than two constraints, consider all possible pairs of constraints as the defining ones in the optimality proof. Can you give a geometric explanation of why some pairs work while others do not?

Exercise 4.2. Draw the feasible region in (x_1, x_2) space defined by the following four constraints:

$$\begin{aligned} x_1 + 3x_2 &\leq 7 \\ x_1 - 4x_2 &\leq 1 \\ -x_1 &\leq 0 \\ -\ x_2 &\leq 0. \end{aligned}$$

Consider (and draw) each of the following additional constraints, (i) to (iv) in turn and answer, for each on, the questions (a) to (c).

(i) $3x_1 - 5x_2 \leq 9$,

(ii) $4x_1 + 19x_2 \leq 34$,

(iii) $2x_1 + 6x_2 \leq 14$,

(iv) $5x_1 - x_2 \leq 31$.

Questions:

(a) Does the feasible region of the additional constraint contain the whole of the feasible region?

(b) Does the additional constraint go through the point of intersection of two of the constraints that define the feasible region? If yes, go to (c), if no, consider the next constraint.

(c) Solve the pair of simultaneous equations to express the additional constraint as a linear combination of the two constraints which intersect on the line of the additional constraint. If the additional constraint coincides with the whole of a single original constraint, you can, of course, express it as the appropriate multiple of that constraint. However, it must also go through the intersection of 2 different pairs of constraints. So, in this case take each of these pairs.
Are the weights nonnegative? Is this result in agreement with your answer to (a)?
Then, consider the next in the list of additional constraints.

Exercise 4.3. Consider the following LP:

$$\begin{array}{lrcl} \text{maximise} & x_1 - x_2 & & \\ \text{subject to} & 3x_1 + 2x_2 & = & 25 \\ & x_1 - x_2 & \geq & -4 \\ & x_1 & \leq & 7 \\ & 2x_1 - x_2 & \leq & 12 \\ & x_1, x_2 & \geq & 0. \end{array}$$

Find the optimal solution by drawing the feasible region and the contours of the function to be maximised, then *prove* the solution to be feasible and optimal.

Chapter 5

Minimisation, nonnegativity, multiple optima

5.1 Minimisation problems

So far we have discussed maximisation problems. When presented with a minimisation problem, we can easily just turn it into a maximisation problem by swapping the signs of the coefficients in the objective function. For example, if the objective is

$$\text{minimise} \quad x_1 - x_2,$$

this is equivalent to solving the maximisation problem

$$\text{maximise} \quad -x_1 + x_2.$$

However, one can solve minimisation problems directly by appropriately modifying the method explained in Section 4.2. The only difference is that, when checking for optimality of a proposed extreme solution, the signs of the dual variables associated with the constraints should be the opposite of that in the case of maximisation problems.

Example. Consider the following LP minimisation problem (note that the constraints of the problem are the same as in the LP (3.1)):

$$\begin{aligned} \text{minimise} \quad & x_1 - x_2 & & \\ \text{subject to} \quad & 2x_1 + x_2 & \leq & \ 10 \\ & x_1 + 2x_2 & \leq & \ 10 \\ & x_1 + x_2 & \leq & \ 6 \\ & x_1 + 3x_2 & \leq & \ 12 \\ & 3x_1 - x_2 & \geq & \ 0 \\ & x_1 + 4x_2 & \geq & \ 4 \\ & x_1, x_2 & \geq & \ 0. \end{aligned} \tag{5.1}$$

The problem is depicted in Figure 5.1.

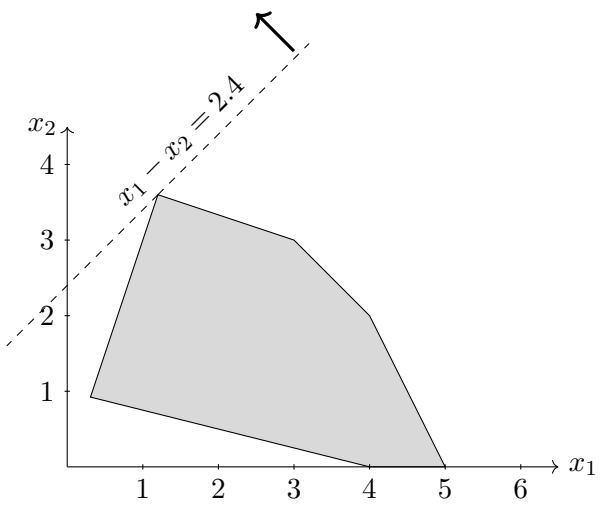


Figure 5.1: Minimisation problem (5.1). The arrow represents the direction of minimisation.

From the diagram, it appears that the optimal solution is the intersection point of constraints 4 and 5. This is point A in Figure 3.2, and we have already determined its coordinates, namely $x_1 = 1.2$ and $x_2 = 3.6$. The value of the objective function at point A is -2.4. In order to show that no feasible point has value less than -2.4, we therefore need to prove that the feasible region is all contained in the half-plane defined by the inequality:

$$x_1 - x_2 \geq -2.4.$$

To do this, as usual we need to obtain the above inequality as a combination of constraints 4 and 5. We need to find dual values y_4 and y_5 such that:

$$\begin{array}{rcl} y_4 \cdot (x_1 + 3x_2 & \leq & 12) \\ y_5 \cdot (3x_1 - x_2 & \geq & 0) \\ \hline x_1 - x_2 & \geq & -2.4. \end{array}$$

We therefore have to solve the system:

$$\begin{aligned} y_4 + 3y_5 &= 1 \\ 3y_4 - y_5 &= -1, \end{aligned}$$

which gives $y_4 = -0.2$, $y_5 = 0.4$. Observe that, since constraint 4 is of type $\leq$ (whereas we need to obtain an inequality of type $\geq$) the multiplier y_4 needs to be nonpositive. Since constraint 5 is of type $\geq$, instead, y_5 needs to be nonnegative. The solution $y_4 = -0.2$, $y_5 = 0.4$ does indeed satisfy the conditions $y_4 \leq 0$ and $y_5 \geq 0$, hence the point $A = (1.2, 3.6)$ is optimal for the LP (5.1).

5.1.1 Summary of the dual variables when minimising

In this table, we summarise the sign that the dual variables must have in order to prove optimality of a feasible extreme solution when minimising.

Constraint	**Dual variable**
$\leq$	nonpositive
$\geq$	nonnegative
$=$	free
Not defining the extreme point	0

5.2 Multiple optimal solutions

It may happen that an objective function reaches it optimum value at more than one extreme point. Consider the LP

$$\begin{array}{lrcl} \text{maximise} & 3x_1 + 3x_2 & & \\ \text{subject to} & x_1 + x_2 & \leq & 5 \\ & 2x_1 - x_2 & \leq & 5 \\ & x_2 & \leq & 3 \\ & x_1, x_2 & \geq & 0, \end{array}$$

which is illustrated in the diagram in Figure 5.2. The optimal extreme solutions are points P, P'. All points in the line segment between P and P' are also optimal, but not extreme. Most computer solvers will find only one of these points. The question of interest is: if we have an optimal solution, can we tell whether or not another optimal point exists?

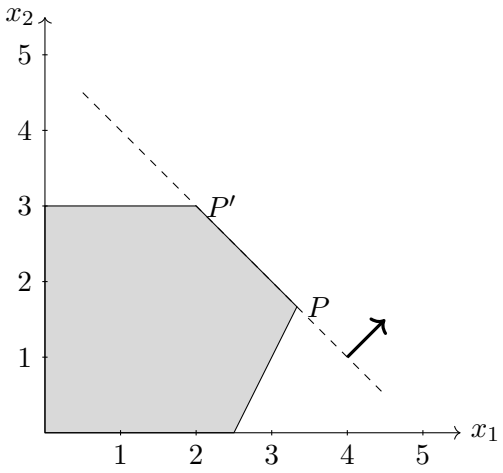


Figure 5.2: A problem with multiple optimal solutions: all points in the line segment between P and P' are optimal.

The point P is defined by constraints 1 and 2. Solving the equations to find the intersection point gives point P as $(\frac{10}{3}, \frac{5}{3})$, and the value of the objective function is 15. To determine the value of the dual variables in order to prove optimality of the extreme point P, we need

to solve the system:

$$\begin{aligned} y_1 + 2y_2 &= 3 \\ y_1 - y_2 &= 3, \end{aligned}$$

which gives $y_1 = 3$, $y_2 = 0$. The dual values of the constraints not defining point P are 0.

The point P' is defined by constraints 1 and 3. Solving the equations to find the intersection point gives point P' as $(2, 3)$, and a value of the objective function of 15. To prove optimality, we need to solve the system:

$$\begin{aligned} y_1 &= 3 \\ y_1 + y_3 &= 3, \end{aligned}$$

which gives $y_1 = 3$, $y_3 = 0$. The dual values of the constraints not defining point P' are 0.

Note that, at either point, the dual value of one of the constraints is 0. Indeed, the only constraint with a positive dual value is constraint 1, which defines the line where both P and P' lie. In general, in order for the problem to have multiple optimal solutions, the following condition must be met.

Necessary condition for multiple optimal solutions. Given an LP problem, let x^* be an optimal extreme solution. If the problem has multiple optimal solutions then some constraint defining x^* must have dual value 0.

In general, the above condition is necessary but not sufficient for multiple optimal solutions to exist. This means that there are examples where some of the constraints defining an extreme optimal solutions have zero dual value, yet there is only one optimum. The following example illustrates this situation.

$$\begin{array}{lrcl} \text{maximise} & 3x_1 + 3x_2 & & \\ \text{subject to} & x_1 + x_2 & \leq & 5 \\ & 2x_1 - x_2 & \leq & 5 \\ & x_2 & \leq & 3 \\ & -x_1 + 2x_2 & \leq & 0 \\ & x_1, x_2 & \geq & 0 \end{array}$$

From Figure 5.3, it appears that the unique optimal solution is the point Q, which lies at the intersection of constraints 1, 2, and 4. To prove optimality of point Q, we need to choose two independent constraints among 1, 2, and 4 as defining constraints.

If we select constraints 1 and 2 as the defining ones, the corresponding dual values are $y_1 = 3$, $y_2 = 0$ (the dual values of all remaining constraints being zero). This proves that point Q is optimal, because $y_1, y_2 \geq 0$, but it would not allow us to conclude that Q is the unique optimum, since constraint 2 defines Q but it has a dual value of 0.

Note that, if instead we selected constraints 2 and 4 as defining constraints, the corresponding dual values would be $y_2 = 3$, $y_3 = 3$ (the dual values of all remaining constraints being zero). This proves that the point Q is optimal, because $y_2, y_3 \geq 0$, but also that there is no other optimal solution, because $y_2, y_3 \neq 0$.

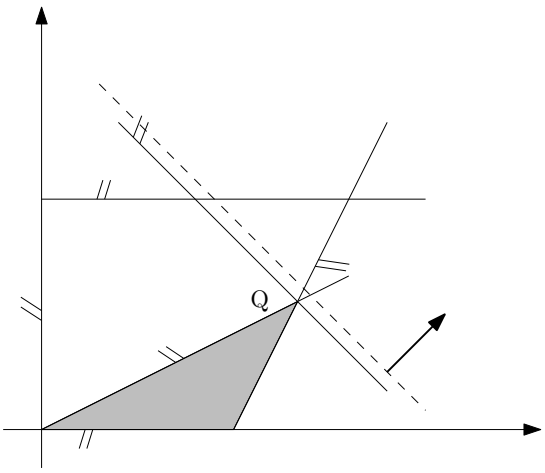


Figure 5.3: A problem with a unique optimum but a dual solution with dual value 0 on some constraint defining the optimal solution.

5.3 Nonnegativities

The approach to proving optimality discussed so far does not distinguish between the resource constraints and the nonnegatives. When all constraints are written in the $\leq$ form, the nonnegatives have a particularly simple structure. The nonnegativity on the variable x_j is a constraint with all zeros except for the jth column where the element is -1. Almost all practical linear programming problems have nonnegative variables, thereby generating as many constraints with this simple structure as there are variables. For example, consider the linear program:

$$\begin{array}{rrcl} \text{maximise} & 3x_1 + 5x_2 + 2x_3 & & \\ \text{subject to} & 2x_1 + 2x_2 + x_3 & \leq & 4 \\ & 3x_1 - x_2 + 2x_3 & \leq & 5 \\ & x_1 - x_2 - x_3 & \geq & 1 \\ & x_1, x_2, x_3 & \geq & 0. \end{array} \tag{5.2}$$

Suppose that we want to decide if the point $x^* = (1.5, 0.5, 0)$ is optimal. Substituting the x values into the constraints given shows that x^* is indeed feasible, and it satisfies at equality the constraints 1, 3, and 6, where constraint 6 is the nonnegativity on x_3. Since the point x^* is feasible and satisfies at equality three constraints, we conclude that x^* is an extreme point. The objective function at this point has value 7.

To decide if x^* is optimal, we need to compute weights y_1, y_3, y_6 such that

$$\begin{array}{rrcl} y_1 \cdot & (\ 2x_1 + 2x_2 + x_3 & \leq & 4)+ \\ y_3 \cdot & (\ x_1 - x_2 - x_3 & \geq & 1)+ \\ y_6 \cdot & (\ x_3 & \geq & 0) \\ \hline = & (3x_1 + 5x_2 + 2x_3 & \leq & 7). \end{array}$$

Since constraint 1 is of type $\leq$ and constraints 3 and 6 are of type $\geq$, we require $y_1 \geq 0$ and

$y_3, y_6 \leq 0$. We need to solve the equations

$$\begin{aligned} 2y_1 + y_3 &= 3 \\ 2y_1 - y_3 &= 5 \\ y_1 - y_3 + y_6 &= 2. \end{aligned}$$

This is a system of 3 variables and 3 constraints. Note, however, that y_6 is the dual variable of the nonnegativity constraint $x_3 \geq 0$. Thus it only appears in the dual equation relative to the coefficients of x_3. That is,

$$y_1 - y_3 + y_6 = 2.$$

For optimality, y_6 has to be nonpositive. However, the conditions $y_1 - y_3 + y_6 = 2$ and $y_6 \leq 0$ are equivalent to $2 - y_1 + y_3 \leq 0$. That is,

$$y_1 - y_3 \geq 2.$$

Thus, once we have the values of y_1 and y_3, testing whether $y_6 \leq 0$ is equivalent to testing whether $y_1 - y_3 \geq 2$, without the need to explicitly compute the value of y_6.

Therefore, in order to decide if x^* is an optimal solution, we only need to solve the system

$$\begin{aligned} 2y_1 + y_3 &= 3 \\ 2y_1 - y_3 &= 5 \end{aligned}$$

and then verify that $y_1 \geq 0$, $y_3 \leq 0$, and $y_1 - y_3 \geq 2$.

The solution is $y_1 = 2$, $y_3 = -1$. Thus $y_1 \geq 0$, $y_3 \leq 0$, and

$$y_1 - y_3 = 3 > 2.$$

It follows that x^* is optimal.

Note that this has allowed us to reduce the number of dual values that we need to compute.

5.3.1 Effective constraints and basic variables

In Section 2.2.1, we distinguished between two types of constraints of an LP: the resource constraints and the nonnegativity constraints. We have seen that an extreme point of an LP in n variables is defined by n independent constraints satisfied at equality at the point. The constraints that define the extreme point can either belong to the resource constraints or to the nonnegativity constraints. This leads to the following definitions.

Consider an extreme point $\bar{x}$ of a system of linear constraints, and a set of constraints defining $\bar{x}$.

Effective constraints. The resource constraints which are defining for $\bar{x}$ are called the *effective constraints* at $\bar{x}$. The remaining resource constraints are *ineffective*.

Basic variables. If a nonnegativity constraint, say constraint $x_j \geq 0$, is defining at $\bar{x}$, then we say that x_j is a *nonbasic variable*. The other variables are *basic variables* at that point.

Observe that nonbasic variables always have value 0. Thus, the only variables that may take nonzero values at the extreme point are the basic ones. (However, it is possible for a basic variable to take 0 value.) Observe also that not all variables need to be defined as nonnegative variables; by definition all variables that are not defined to be nonnegative are therefore basic.

Given that an extreme point is defined by n independent constraints, the number of nonbasic variables plus the number of effective constraints must be n. Considering that the number of basic variables is n minus the number of nonbasic variables, it follows that:

> At any extreme point, the number of effective constraints equals the number of basic variables.

For example, for the LP in (5.2), at point x^* of coordinates $x_1 = 1.5$, $x_2 = 0.5$, $x_3 = 0$, the effective constraints are constraints 1 and 3, while the basic variables are x_1 and x_2. Constraint 2 is ineffective, while variable x_3 is nonbasic.

5.3.2 Optimality conditions and nonnegatives

Consider a maximisation LP with n nonnegative variables and m resource constraints.

$$
\begin{array}{lrcl}
\text{maximise} & c_1x_1 + c_2x_2 + \ldots + c_nx_n & & \\
\text{subject to} & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \gtrless & b_1 \\
& a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \gtrless & b_2 \\
& \vdots & & \\
& a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \gtrless & b_m \\
& x_1, x_2, \ldots, x_n & \geq & 0
\end{array}
$$

Suppose we are given an extreme point x^*, and we want to decide if the point is optimal. Let γ^* be the value of the objective function at x^*. We need to assign dual values to the resource constraints and to the nonnegativity constraints. Let $y_1, \ldots, y_m$ be the dual variables corresponding to the m resource constraints, and $y_{m+1}, y_{m+2}, \ldots, y_{m+n}$ the dual variables corresponding to the nonnegativity constraints $x_1 \geq 0$, $x_2 \geq 0$, $\ldots, x_n \geq 0$. The multipliers need to satisfy the following:

$$
\begin{array}{rrcccccl}
y_1 \cdot & (\, a_{11}x_1 + & a_{12}x_2 + & \ldots & + a_{1n}x_n & \gtrless & b_1) \\
y_2 \cdot & (\, a_{21}x_1 + & a_{22}x_2 + & \ldots & + a_{2n}x_n & \gtrless & b_2) \\
\vdots & & & & & & \\
y_m \cdot & (a_{m1}x_1 + & a_{m2}x_2 + & \ldots & + a_{mn}x_n & \gtrless & b_m) \\
y_{m+1} \cdot & (\quad x_1 & & & & \geq & 0) \\
y_{m+2} \cdot & (\quad & x_2 & & & \geq & 0) \\
\vdots & & & & & & \\
y_{m+n} \cdot & (\quad & & & x_n & \geq & 0) \\
\hline
= & (\, c_1x_1 + & c_2x_2 + & \ldots & + c_nx_n & \leq & \gamma^*).
\end{array}
$$

The dual values of the ineffective constraints are zero, while dual values of the effective constraints are required to be of the appropriate sign for the point to be an optimum.

If a variable x_j is basic, then the dual value corresponding to the constraint $x_j \geq 0$ should be 0.
If a variable x_j is nonbasic, then the dual value corresponding to the constraint $x_j \geq 0$ should be nonpositive.

For every $j = 1, \ldots, n$, the dual variables should satisfy the following equation

$$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m + y_{m+j} = c_j. \tag{5.3}$$

Note that, if x_j is a basic variable, then y_{m+j} should be zero, thus (5.3) is equivalent to requiring

$$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m = c_j.$$

If x_j is nonbasic, then y_{m+j} should be less than or equal to zero, thus (5.3) is equivalent to requiring

$$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m \geq c_j.$$

Therefore, it is not necessary to consider explicitly the dual variables relative to nonnegativity constraints when checking for optimality of an extreme point. The conditions for optimality for maximisation problems are summarised in Table 5.1.

Resource constraint	**Dual variable**
$\leq$	$\geq$
$\geq$	$\leq$
$=$	free
Not effective	0
Variable x_j	**Dual condition**
Basic	$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m = c_j$
Nonbasic	$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m \geq c_j$

Table 5.1: Optimality conditions for **maximisation problems**.

Minimisation problems. Analogous optimality conditions hold for minimisation problems. We summarise them in Table 5.2.

5.3.3 Multiple optimal solutions

We can specialise the necessary conditions for multiple optimal solutions given in Section 5.2 to the case where all variables are nonnegative as follows.

Resource constraint	**Dual variable**
$\leq$	$\leq$
$\geq$	$\geq$
$=$	free
Not effective	0
Variable x_j	**Dual condition**
Basic	$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m = c_j$
Nonbasic	$a_{1j}y_1 + a_{2j}y_2 + \cdots + a_{mj}y_m \leq c_j$

Table 5.2: Optimality conditions for **minimisation problems.**

Necessary conditions for multiple optimal solutions

At an extreme optimal solution x^*, for a given set of defining constraints for x^*, let y^* be the corresponding dual values. If the problem has multiple optimal solutions then (at least) one of the following must occur:

1. The dual value associated with some effective constraint is 0;
2. There is some nonbasic variable x_j such that $a_{1j}y_1^* + \cdots + a_{mj}y_m^* = c_j$.

Indeed, according to Section 5.2, a necessary condition in order to have multiple optima is that some of the constraints defining x^* have zero dual value. If the constraint having zero dual value is an effective constraint, then we have condition 1 above. If the constraint having zero dual value is a nonnegativity constraint associated to a nonbasic variable, say constraint $x_j \geq 0$, then we have that $y^*_{m+j} = 0$. By (5.3), this implies that $a_{1j}y_1^* + \cdots + a_{mj}y_m^* = c_j$, which is condition 2 above.

5.4 Exercises

Exercise 5.1. Consider the following LP:

$$\begin{array}{lrcl} \text{minimise} & x_1 + 2x_2 & & \\ \text{subject to} & x_2 & \geq & 2 \\ & 5x_1 + 3x_2 & \geq & 30 \\ & x_1 & \leq & 6 \\ & x_1, x_2 & \geq & 0. \end{array}$$

(a) *Find* the optimal solution by drawing the feasible region and the contours of the objective function, and then prove the solution to be both feasible and optimal.

(b) What is the solution if the objective function is *maximise* $x_1 + 2x_2$ subject to the same constraints?

(c) What is the solution if the constraint $x_2 \geq 2$ becomes $x_2 = 2$?

(d) What is the solution if the constraint $x_2 \geq 2$ is removed?

Exercise 5.2. Consider the following LP:

$$\begin{array}{lrcl} \text{maximise} & 3x_1 - 6x_2 & & \\ \text{subject to} & -x_1 + x_2 & \leq & 2 \\ & 5x_1 + 8x_2 & \leq & 42 \\ & x_1 - 2x_2 & \leq & 3 \\ & 3x_1 + 2x_2 & \leq & 21 \\ & x_1, x_2 & \geq & 0. \end{array} \tag{5.4}$$

(a) Does the problem have multiple optimal solutions?

(b) Suppose that we replace the objective function by *maximise* $6x_1+4x_2$. Are the necessary conditions for the existence of multiple optimal solutions satisfied? Does the problem have multiple optimal solutions in this case?

Exercise 5.3. Consider the following LP:

$$\begin{array}{lrcl} \text{maximise} & 3x_1 + 4x_2 - 4x_3 & & \\ \text{subject to} & x_1 + x_2 + x_3 & \leq & 2 \\ & 2x_1 - x_2 - 2x_3 & \geq & 1 \\ & x_2 - 3x_3 & \leq & 3 \\ & x_1, x_2, x_3 & \geq & 0. \end{array}$$

(a) Prove that the point $x_1 = 5/4, x_2 = 0, x_3 = 3/4$ is not optimal.

(b) Prove that the point $x_1 = 1, x_2 = 1, x_3 = 0$ is optimal.

Exercise 5.4. *Solve the LP* (5.4) *in Exercise 12.2 using the Excel solver.*

Chapter 6

The dual program

Thus far, we have introduced the concept of dual values as a means to prove optimality of LP problems. In this lecture, we will see that the dual values can themselves be seen as optimal solutions of another LP problem, called the dual problem.

6.1 Dual problem - a numerical example

Consider the maximisation problem we introduced in Chapter 3

$$\begin{array}{lrcl} \text{maximise} & 8x_1 + 3x_2 & & \\ \text{subject to} & 2x_1 + x_2 & \leq & 10 \\ & x_1 + 2x_2 & \leq & 10 \\ & x_1 + x_2 & \leq & 6 \\ & x_1 + 3x_2 & \leq & 12 \\ & -3x_1 + x_2 & \leq & 0 \\ & -x_1 - 4x_2 & \leq & -4 \\ & x_1,\, x_2 & \geq & 0. \end{array} \tag{6.1}$$

We associate to the above the following minimisation problem

$$\begin{array}{lrcl} \text{mininimise} & 10y_1+10y_2+6y_3+12y_4+0y_5-4y_6 & & \\ \text{subject to} & 2y_1+ y_2+ y_3+ y_4-3y_5- y_6 & \geq & 8 \\ & y_1+ 2y_2+ y_3+ 3y_4+ y_5-4y_6 & \geq & 3 \\ & y_1,\, y_2,\, y_3,\, y_4,\, y_5,\, y_6 & \geq & 0. \end{array} \tag{6.2}$$

Note the structure of the minimisation problem:

- The minimisation problem has one nonnegative variable for every resource constraint of the maximisation problem. The coefficient of each variable in the objective function of the minimisation problem is the right-hand-side of the corresponding constraint in the maximisation problem.
- The minimisation problem has a resource constraint for every variable of the maximisation problem, and the right-hand-side of each constraint in the minimisation problem is the coefficient of the corresponding variable in the objective function of the maximisation problem.

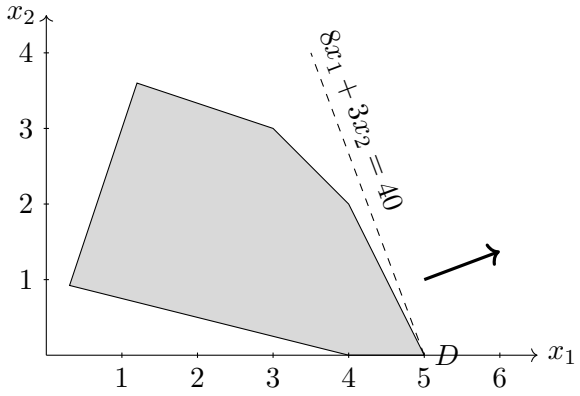


Figure 6.1: Representation of Problem (6.1).

- The coefficients in the rows of the minimisation problem are the coefficients in the columns of the maximisation problem, and vice versa.

What is the relationship between the solutions to the two problems? Let us look at this from two different perspectives:

From the point of view of the maximisation problem. From the diagram in Figure 6.1 it would appear that the point D is the optimum. At point D, the only effective constraint is constraint 1, while x_1 is a basic variable and x_2 is a nonbasic variable. It follows that point D is the point of coordinates $x_1 = 5$, $x_2 = 0$. The value of the objective function at point D is 40. One can check that the point is indeed feasible. To prove optimality, we have to find dual values $y_1^*, \ldots, y_6^*$, satisfying the conditions in Table 5.1. Since constraints 2, ..., 6 are ineffective, we must have $y_2^*, \ldots, y_6^* = 0$. Since x_1 is basic, y_1^* should satisfy

$$2y_1^* + 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 0 - 3 \cdot 0 - 1 \cdot 0 = 8.$$

Clearly the only choice is $y_1^* = 4$. Furthermore, we need to check that $y_1^* = 4 \geq 0$ and that

$$y_1^* + 2 \cdot 0 + 1 \cdot 0 + 3 \cdot 0 + 1 \cdot 0 - 4 \cdot 0 = 4 \geq 3.$$

It follows that $x_1^* = 5$, $x_2^* = 0$ *is the optimal solution of the maximisation problem* (6.1)*, and the optimal value is* 40.

From the point of view of the minimisation problem. Note that the point y^* of coordinates $y_1^* = 4$, $y_2^*, \ldots, y_6^* = 0$, as computed above, is feasible for the minimisation problem (6.2). The value of the objective function at this point is $10 \cdot 4 = 40$. At point y^*, the only effective constraint is constraint 1, while the only basic variable is y_1. The number of

basic variables and effective constraints are the same, indicating that y^* is an extreme point for the minimisation LP (6.2). Is y^* optimal?

In order to verify optimality, we need to find dual values, say ω_1, ω_2, for the two constraints of (6.2), so that they satisfy the conditions in Table 5.2. Since constraint 2 of the minimisation LP (6.2) is ineffective, it should be $\omega_2 = 0$. Since y_1 is basic, ω_1 should satisfy

$$2\omega_1 + 1 \cdot \omega_2 = 10.$$

The only solution is $\omega_1 = 5$. We finally need to check that $\omega_1 = 5 \geq 0$, and that ω_1 satisfies the following conditions, corresponding to the nonbasic variables $y_2, \ldots, y_6$:

$$\begin{array}{rcrl} \omega_1 + 2 \cdot 0 & = & 5 & \leq 10 \\ \omega_1 + 1 \cdot 0 & = & 5 & \leq 6 \\ \omega_1 + 3 \cdot 0 & = & 5 & \leq 12 \\ -3\omega_1 + 1 \cdot 0 & = & -15 & \leq 0 \\ -\omega_1 - 4 \cdot 0 & = & -5 & \leq -4. \end{array}$$

It follows that y^* *is the optimal solution of the minimisation problem* (6.2)*, and the optimal value is* 40.

Note that the dual values $\omega_1 = 5$, $\omega_2 = 0$ proving the optimality of y^* are exactly the coordinates of the optimal solution of the maximisation problem (6.1).

This example shows that by finding the optimal solution to the maximisation problem, we have simultaneously solved the minimisation problem, and conversely, by finding the optimal solution to the minimisation problem, we have simultaneously solved the maximisation problem. Furthermore, both problems have optimal value equal to 40. The problems are *dual to each other*; solving one solves the other.

6.2 Dual problems - general statement

For a matrix $A \in \mathbb{R}^{n \times m}$, vectors $b \in \mathbb{R}^n$, and $c \in \mathbb{R}^m$, consider the maximisation problem:

$$\begin{array}{rl} \text{maximise} & c^\top x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

The *dual problem* is

$$\begin{array}{rl} \text{minimise} & b^\top y \\ \text{subject to} & A^\top y \geq c \\ & y \geq 0 \end{array}$$

where $A^\top$, $b^\top, c^\top$ represent, respectively, the transposes of A, b, and c. Note that if we start from an LP problem and we take the dual of its dual, we get back to the original problem.

The original problem is often referred to as the *primal*. Primal and dual problems have the following relation:

- If the primal problem is a maximisation problem, the dual is a minimisation problem. If the primal is a minimisation problem, the dual is a maximisation problem.

- For every variable in the primal problem, there is a constraint in the dual.
- For every constraint in the primal, there is a variable in the dual.
- The objective function coefficients in the primal are the right-hand-side coefficients in the dual, and vice versa.

In general, we can take the dual of problems with resource constraints of various types, $\leq$, $\geq$, or $=$, and with variables that can be restricted to be nonnegative, nonpositive, or free. The table below summarises the relationship between the constraints of the primal and the variables of the dual, as well as between the variables of the primal and the constraints of the dual.

Primal (Dual) maximising	**Dual (Primal) minimising**
constraint $\leq$	variable ≥ 0
constraint $\geq$	variable ≤ 0
constraint $=$	variable free
variable ≥ 0	constraint $\geq$
variable ≤ 0	constraint $\leq$
variable free	constraint $=$

Consider the following minimisation problem as an example:

$$\begin{array}{rrcl} \text{minimise} & x_1 - 2x_2 - x_3 & & \\ \text{subject to} & -x_1 + x_2 + 2x_3 & = & 4 \\ & 2x_1 \quad - x_3 & \geq & -2 \\ & -2x_1 - x_2 + x_3 & \leq & 1 \\ & x_1 \leq 0,\ x_3 \geq 0 & & \end{array}$$

Since the problem is a minimisation problem, the dual is a maximisation problem. The problem has three resource constraints, thus the dual will have three variables, y_1, y_2, y_3. The problem has three variables, thus the dual will have three resource constraints. The signs of the dual variables ("$\leq$", "$\geq$", or "free") and the directions of the dual constraints are obtained according to the previous table, reading it from right to left. For example, x_1 is a variable of type "$\leq$" in a minimisation problem, therefore the corresponding dual constraint is of type "$\geq$". The dual is the following:

$$\begin{array}{rrcl} \text{maximise} & 4y_1 - 2y_2 + y_3 & & \\ \text{subject to} & -y_1 + 2y_2 - 2y_3 & \geq & 1 \\ & y_1 \quad - y_3 & = & -2 \\ & 2y_1 - y_2 + y_3 & \leq & -1 \\ & y_2 \geq 0,\ y_3 \leq 0 & & \end{array}$$

One can verify that, if we write the dual of the above problem, we end up with the original one.

6.2.1 Duality and optimality

In the example we have seen in the previous section, the primal problem and its dual have the same optimal value. It can be shown that this fact holds in general. This is known as the Strong Duality Theorem of Linear Programming.

Strong Duality Theorem. If a Linear Programming problem admits an optimal solution, then also its dual admits an optimal solution. Furthermore, the optimal values of the primal problem and of its dual coincide.

This is the most important result in the theory of Linear Programming, and it has applications in several areas beyond operational research, such as economics and game theory.

We can now re-cast the optimality conditions for an extreme solution – presented in Section 5.3.2 and summarised in Tables 5.1 and 5.2 – in the context of duality. Namely:

Optimality conditions for an extreme solution. An extreme solution x^* to an LP is optimal if and only if there exists a feasible solution y^* for the dual that satisfies the following conditions:

- y^* satisfies at equality all the dual constraints corresponding to basic variables.
- For every ineffective constraint for x^*, the corresponding component of y^* is 0.

Furthermore, such a solution y^* is optimal for the dual.

For example, for the LP (6.1), the extreme solution $x^* = (5, 0)$ is optimal because we can find a point $y^* = (4, 0, 0, 0, 0, 0)$ that is feasible for the primal, satisfies at equality the dual constraints corresponding to the basic variable x_1, and is 0 in all the components that correspond to the ineffective constraints (constraints $2, \ldots, 6$).

6.3 An economics interpretation of the dual

When formulating a real-world problem as an LP, the variables, constraints, and objective function of the LP typically have a clear interpretation in terms of the original problem. It is often the case that also the variables, constraints, and objective function of the dual can be interpreted and give further information on the original problem. A typical case when such an interpretation of the dual is possible is in the case of resource allocation problems. Here we give an example of such a problem.

A chip's manufacturer produces four types of memory chips in one their state-of-the-art factories. These chips are sold to electronics corporations for their devices. The main resources used in the chip's production are labor and silicon wafers. The factory's problem for the next month is:

$$
\begin{array}{lrcll}
\text{maximise} & 15x_1 + 24x_2 + 32x_3 + 40x_4 & & & \\
\text{subject to} & x_1 + 2x_2 + 8x_3 + 7x_4 & \leq & 2000 & \text{(Labour)} \\
& 6x_1 + 8x_2 + 12x_3 + 15x_4 & \leq & 15000 & \text{(Silicon wafers)} \\
& x_1,\, x_2,\, x_3,\, x_4 & \geq & 0 &
\end{array}
$$

where variable x_j represents the number of chips produced, in thousands, the objective function coefficients represent the profit (in £ per unit) of the four types of chips, 2000 is the number of hours of labour available, and 15000 is the number of silicon wafers available. Therefore, the objective function's coefficients represent thousand £ per unit of each variable $x_1, \ldots, x_4$. (For example, $x_2 = 1$ corresponds to producing 1000 chips of type 2, which requires 2 hours of labour, 8 wafers, and gives a profit of £24000, that is, $24x_2$ thousand £.)

Mango Inc., a giant consumer electronics corporation, urgently needs as many units as possible of a new type of memory chip, due to a stronger-than-expected demand for their new smart phone. Mango Inc. would like the chip manufacturer to devote all resources of the factory to the production the new type of chip. Mango Inc. intends to determine prices to offer for each of the resources in order to convince the manufacturer to sell them, while minimising the total sum paid.

Let y_1 denote the price (in thousand £) that the buyer intends to pay per hour of labour, and let y_2 denote the price (in thousand £) that Mango Inc. intends to pay for each silicon wafer. Clearly, these prices must be greater than or equal to zero. That is, $y_1, y_2 \geq 0$.

In order to persuade the manufacturer to sell, the buyer must offer prices for its resources such that the manufacturer is not tempted to retain its resources to produce chips of type 1. So the buyer must offer prices such that the total value of the resources used in producing chips of type 1 is at least the amount of profit the manufacturer could attain. That is,

$$y_1 + 6y_2 \geq 15.$$

Similarly, for chips of type 2 Mango Inc. must pitch their prices so that the manufacturer is at least as well off selling as he would be by not selling:

$$2y_1 + 8y_2 \geq 24.$$

Each type of chip generates such a constraint. On the other hand, the buyer wants to pay the minimum amount possible for the entire amount of resources, that is, it wants to minimise $2000y_1 + 15000y_2$. Mango Inc.'s problem is therefore

$$\begin{array}{lrcl} \text{minimise} & 2000y_1 + 15000y_2 & & \\ \text{subject to} & y_1 + 6y_2 & \geq & 15 \\ & 2y_1 + 8y_2 & \geq & 24 \\ & 8y_1 + 12y_2 & \geq & 32 \\ & 7y_1 + 15y_2 & \geq & 40 \\ & y_1,\, y_2 & \geq & 0. \end{array}$$

The manufacturer's and the Mango Inc.'s problems are dual to each other. Under this interpretation, the dual values represent prices of resources (which explains why dual values are often also called "shadow prices"). Of course, we know that the total amount that the buyer will have to pay is the same as the amount of profit that the manufacturer would achieve by carrying on its usual production. The interpretation of dual variables as prices will be further explained in Chapter 7

6.4 Exercises

Exercise 6.1. A campsite owner, Brown, wants to expand his site and therefore needs to persuade a neighbouring small farmer, Smith, to both rent him 30 acres of land and to work for him 20 weeks a year. He wants to determine the prices (rent of the land in £ per acre and wages of Smith in £ per man-week, x_1 and x_2, respectively) he should offer in order to minimise his total payment to Smith which is $30x_1 + 20x_2$. The constraints which limit his cost minimisation are the opportunities which Smith has to use the land and labour himself. In other words, the prices of rent and labour must be such as to ensure that Smith cannot earn more revenue (by using the land and labour necessary to produce any of the three different crops that are available to him) than by selling them to Brown. The data for the crops available to Smith are shown below.

Crop	Revenue (£ per tonne)	Land requirement (acres per tonne)	Labour requirement (man-weeks per tonne)
1	7	10	3
2	5	6	2
3	3	2	2

(a) Formulate and solve Brown's problem as an LP.

(b) Formulate and solve a problem dual to (a). How can you interpret this LP?

Exercise 6.2. Consider the following LP:

$$\begin{array}{lrcl} \text{minimise} & 2x_1 + 4x_2 - 7x_3 & & \\ \text{subject to} & -2x_1 + 4x_2 + 3x_3 & \leq & 20 \\ & -x_1 - x_2 + x_3 & = & 8 \\ & 3x_1 + 5x_2 + 4x_3 & \geq & 0 \\ & x_1 & \leq & 0 \\ & x_2,\, x_3 & \geq & 0. \end{array}$$

(a) Write the dual of this LP.

(b) Verify that the proposed solution $x^* = (-4, 0, 4)$ is optimal, by finding the appropriate dual solution.

Exercise 6.3. Consider the following LP:

$$\begin{array}{lrcl} \text{maximise} & 4x_1 - 9x_2 & & \\ \text{subject to} & -2x_1 + 9x_2 & \leq & 54 \\ & x_1 + 3x_2 & \leq & 33 \\ & 5x_1 + 5x_2 & \leq & 120 \\ & x_1 - x_2 & \leq & 19 \\ & 5x_1 - 14x_2 & \leq & 140 \\ & x_1 - 6x_2 & \leq & 18 \\ & x_1,\, x_2 & \geq & 0 \end{array}$$

(a) Write the dual of this LP.

(b) Solve the problem via the Excel solver. What are the allowable increase and decrease for constraint 6?

(c) Draw the feasible region, and illustrate the allowable increase and decrease for constraint 6 on the diagram.

(d) What is the relationship between the coefficients of constraints 4 and 6, and the allowable increase and decrease of the objective coefficient of variable 1?

Chapter 7

Sensitivity analysis

One of the weaknesses of Linear Programming is the underlying assumption that all the coefficients are known with certainty. It is a deterministic model of the real situation. However, often important characteristics of the real-world problem depend on random eventualities (for example, demand, prices, resource availability, etc.).

A useful feature of Linear Programming is that one can analyse how sensitive the optimal solution is to changes in the data. That means that one can evaluate, without the need to re-solve the LP, how changing the values of certain coefficients affects the optimal solution and its value in the objective function.

There are two main limitations that one should however consider: sensitivity analysis can only provide information on the impact of changing one coefficient at the time, and only for changes within a certain range (which depends on the problem and on the specific coefficient).

7.1 The dual values as marginal values

Consider the problem:

$$\begin{array}{rrcl} \text{maximise} & 2x_1 + 8x_2 & & \\ \text{subject to} & 2x_1 + x_2 & \leq & 10 \\ & x_1 + 2x_2 & \leq & 10 \\ & x_1 + x_2 & \leq & 6 \\ & x_1 + 3x_2 & \leq & 12 \\ & -3x_1 + x_2 & \leq & 0 \\ & -x_1 - 4x_2 & \leq & -4 \\ & x_1, x_2 & \geq & 0. \end{array}$$

In Section 4.1.2, it was shown that the optimum solution to this problem is at the extreme point x^* defined by constraints 4 and 5, which is the point of coordinates $x_1 = 1.2$, $x_2 = 3.6$. The maximum objective function value is 31.2. The dual values of the effective constraints (computed in Section 4.1.2) are $y_4^* = 2.6$ and $y_5^* = 0.2$.

How does the optimal value change if we were to change the value of the right-hand-side of constraint 5 by some amount θ, from 0 to $0 + \theta$? (Here, θ could be also negative.) The

new problem will be:

$$\begin{array}{rrcl} \text{maximise} & 2x_1 + 8x_2 & & \\ \text{subject to} & 2x_1 + x_2 & \leq & 10 \\ & x_1 + 2x_2 & \leq & 10 \\ & x_1 + x_2 & \leq & 6 \\ & x_1 + 3x_2 & \leq & 12 \\ & -3x_1 + x_2 & \leq & \theta \\ & -x_1 - 4x_2 & \leq & -4 \\ & x_1, x_2 & \geq & 0. \end{array}$$

Figure 7.1 represents the original and the modified problems.

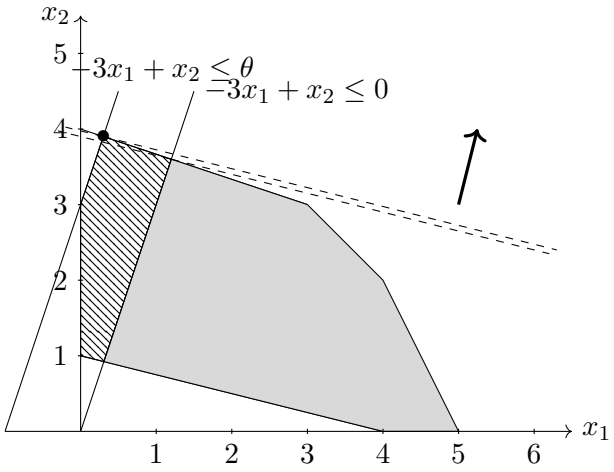


Figure 7.1: Feasible regions of the original and modified problem, and corresponding optimal contours of the objective function. The new optimal solution is indicated by the black dot. Note that the optimal solution changes, but it is still defined by constraints 4 and 5. (The diagram corresponds to the value $\theta = 3$.)

If the change amount θ is too large, there is not much one can say without re-solving the problem. However, let us assume that the change is "small enough" that the optimal solution will still be defined by constraints 4 and 5. (We will later discuss how small θ needs to be in order to satisfy this assumption.)

Denote by γ^* the optimal value of the modified problem. Assuming that the optimum point is still defined by constraints 4 and 5, in order to compute the dual values of constraints 4 and 5, we need compute multipliers y_4 and y_5 such that:

$$\begin{array}{rcl} y_4 \cdot (\ x_1 + 3x_2 & \leq & 12) \\ y_5 \cdot (-3x_1 + x_2 & \leq & \theta) \\ \hline 2x_1 + 8x_2 & \leq & \gamma^*. \end{array}$$

We must solve the equations:

$$\begin{aligned} y_4 - 3y_5 &= 2 \\ 3y_4 + y_5 &= 8. \end{aligned}$$

Note that the coefficients of this system of equations are unchanged, thus the equations are exactly the same ones we solved in Section 4.1.2. The dual values are therefore the same as before, namely $y_4 = 2.6$, $y_5 = 0.2$.

It follows that the value of γ^* is $12 \cdot 2.6 + \theta \cdot 0.2 = 31.2 + 0.2 \cdot \theta$. Therefore, a change by θ in the right-hand-side of constraint 5 leads to a change of $0.2 \cdot \theta$ in the optimal value, where 0.2 is the dual value of constraint 5.

This relationship is true in general. The size and sign of a dual value gives the rate of change of the objective function value as the right-hand-side of a constraint changes. In symbols,

$$y_i = \frac{\partial(\text{optimal value})}{\partial(b_i)}.$$

Notice that the change in resource availability can be either up or down. If the i^{th} constraint is a $\leq$ and the problem is maximisation so that the optimum dual value y_i is nonnegative, then, if the resource increases, so will the objective function; and if the resource decreases, then so will the objective function. This is to be expected, since increasing the right-hand-side of a $\leq$ constraint makes the problem less constrained, and therefore there could be solutions with higher value, whereas decreasing the right-hand-side makes the problem more constrained, and so the previous optimal solution might become infeasible and the new optimum would have lower value.

Analogously, if the constraint is a $\geq$ and the problem is maximisation, so that the optimum the dual value y_i of the i^{th} constraint is nonpositive, then, if the right-hand-side increases, the optimal objective value will decrease, and if the resource decreases, then the optimal objective value will increase.

Of course, if the dual variable has value 0, i.e., $y_i = 0$, then for small enough changes in the right-hand-side of the i^{th} constraint, there will be no change in the value of the optimal objective value

7.1.1 Ranges for the coefficients of the right-hand-side

The interpretation of the dual value as the rate of change of the objective function that results from a change in a resource is only true for a limited range of values of the right-hand-side of the constraint. The derivation of the ranges is easy for the ineffective constraints. The general derivation of this range for effective constraints is outside the scope of these lectures, however we will compute it for the previous example and give a geometric intuition through diagrams.

All LP solvers output the ranges for the right-hand-sides. In the Excel Solver output, these can be found in the Sensitivity Report, under the headings Allowable Increase and Allowable Decrease.

Ineffective constraints. If the ineffective constraint is a $\leq$ constraint, then clearly the right-hand-side can increase to infinity without affecting the solution. The right-hand-side can decrease until it is low enough for the constraint to be satisfied at equality, that is, to the value below which the current solution would be infeasible. Similarly, the right-hand-side of

an ineffective $\geq$ constraint could decrease to minus infinity and increase to the value above which the current solution will be infeasible.

For example, consider the LP (7.1). At the optimal solution $x_1 = 1.2$, $x_2 = 3.6$, the constraints 1, 2, 3, and 6 are all ineffective. By how much can we change the right-hand-side of these constraints before the optimal solution changes? For constraint 1, we have that

$$2 \cdot 1.2 + 1 \cdot 3.6 = 6.$$

Since the right-hand-side is 10, this can decrease to 6 without affecting the optimal solution, and it can increase to infinity. That is, the optimal solution $x_1 = 1.2$, $x_2 = 3.2$ does not change as long as the right-hand-side b_1 of the first constraint is within the range $6 \leq b_1 \leq +\infty$.

Analogously, we can compute the ranges for the other constraints:

$$\begin{array}{lrcll} \text{Constraint 2:} & 1 \cdot 1.2 + 2 \cdot 3.6 & = & 8.4 & \Longrightarrow \quad 8.4 \leq b_2 \leq +\infty, \\ \text{Constraint 3:} & 1 \cdot 1.2 + 1 \cdot 3.6 & = & 4.8 & \Longrightarrow \quad 4.8 \leq b_3 \leq +\infty, \\ \text{Constraint 6:} & -1 \cdot 1.2 - 4 \cdot 3.6 & = & -15.6 & \Longrightarrow \quad -15.6 \leq b_6 \leq +\infty. \end{array}$$

Effective constraints. Let us consider the previous example (7.1), where we change the right-hand-side of constraint 5.

From the diagram on the left in Figure 7.2, it is apparent that if we increase the right-hand-side of constraint 5, the optimal solution of the problem remains defined by constraints 4 and 5 until the border of constraint 5 passes through the intersection of constraint 4 and the nonnegativity of x_1.

One can compute that the intersection point of constraint 4 and $x_1 = 0$ is the point $(0, 4)$. Constraint 5 passes through this point when its right-hand-side becomes

$$-3 \cdot 0 + 1 \cdot 4 = 4.$$

Similarly, from the diagram on the right of Figure 7.2, it is apparent that if we decrease the right-hand-side of constraint 5, the optimal solution of the problem remains defined by constraints 4 and 5 until the border of constraint 5 passes through the intersection of constraints 3 and 4 (which is point B in Figure 3.2).

One can compute that point B has coordinates $(3, 3)$. Constraint 5 passes through this point when its right-hand-side becomes

$$-3 \cdot 3 + 1 \cdot 3 = -6.$$

It follows that the range for the right-hand-side of constraint 5 is:

$$-6 \leq b_5 \leq 4.$$

7.2 Coefficients of the objective function

In this section, we study how changes in one of the objective function coefficients affect the value of the optimal solution. As before, if the coefficient falls within a certain range, the new optimal objective value can be computed without having to re-solve the problem. Computing this range is a straightforward task if the change occurs to the coefficient of a variable that is nonbasic, while it is more complicated for basic variables.

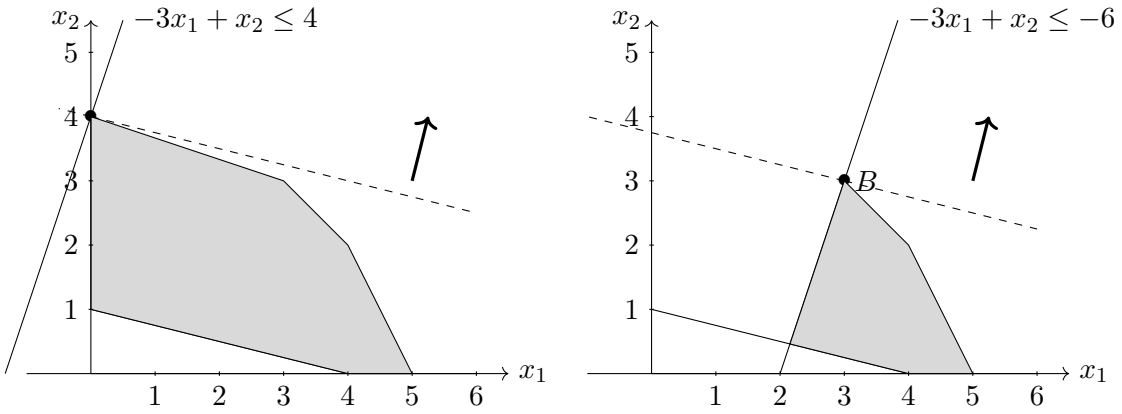


Figure 7.2: Representation of the largest and smallest values that the right-hand-side of constraint 5 can take in order for constraints 4 and 5 to remain effective.

7.2.1 Nonbasic variables

If the problem is a maximisation and a variable x_j is nonbasic, then clearly its coefficient c_j in the objective function can decrease to minus infinity and the variable x_j will continue at the value zero. As the coefficient c_j of the objective function increases, there will be a value at which the solution will be multiply optimal, and then above that the current solution will be suboptimal. The new optimal solution will have the variable x_j at a nonzero value.

Consider the LP problem (6.1), discussed in Section 6.1:

$$\begin{array}{lrcl} \text{maximise} & 8x_1 + 3x_2 & & \\ \text{subject to} & 2x_1 + \ x_2 & \leq & 10 \\ & x_1 + 2x_2 & \leq & 10 \\ & x_1 + \ x_2 & \leq & 6 \\ & x_1 + 3x_2 & \leq & 1 \\ & -3x_1 + \ x_2 & \leq & 0 \\ & -x_1 - 4x_2 & \leq & -4 \\ & x_1, \, x_2 & \geq & 0. \end{array}$$

At the optimal solution, x_1 is basic with a value of 5, and x_2 is nonbasic. The effective constraint is constraint 1 with dual value $y_1 = 4$. Suppose we replace the objective function coefficient of the nonbasic variable x_2 with another number c_2. Since all other coefficients in the LP are unchanged, in order to check optimality of the solution $x_1 = 5$, $x_2 = 0$ we only need to confirm that the dual constraint relative to the variable x_2 remains satisfied by the dual solution. That is,

$$y_1 + 2y_2 + y_3 + 3y_4 + y_5 - 4y_6 \geq c_2.$$

Thus, we need to confirm that

$$1 \cdot 4 + 2 \cdot 0 + 1 \cdot 0 + 3 \cdot 0 + 1 \cdot 0 - 4 \cdot 0 = 4 \geq c_2.$$

The solution would be multiply optimal if the value of c_2, now 3, were to increase to 4. Beyond the value of 4, the solution would be non-optimal. Thus the upper limit of the value of c_2 is 4 and its lower limit is $-\infty$.

For a maximisation LP problem. Given an optimal extreme point x^*, let y^* be an optimal dual solution. Let x_j be a nonbasic variable for x^*. The solution x^* remains optimal if we change the objective function coefficient c_j within the range

$$-\infty \leq c_j \leq a_{1j}y_1^* + \cdots + a_{mj}y_m^*.$$

An analogous statement holds for minimisation problems.

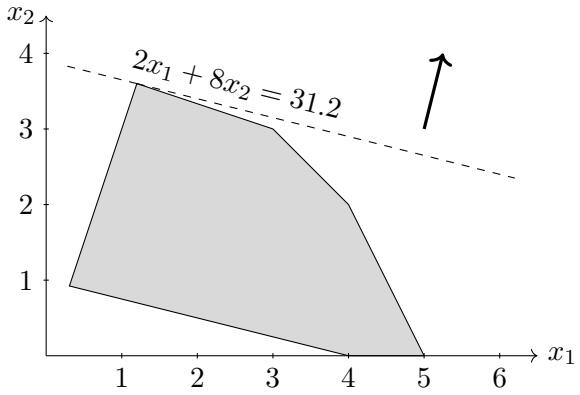
For a minimisation LP problem. Given an optimal extreme point x^*, let y^* be an optimal dual solution. Let x_j be a nonbasic variable for x^*. The solution x^* remains optimal if we change the objective function coefficient c_j within the range

$$a_{1j}y_1^* + \cdots + a_{mj}y_m^* \leq c_j \leq +\infty.$$

Note: In Excel Solver's sensitivity report, these quantities are displayed as the allowable increase/decrease for variable cells for nonbasic variables.

7.2.2 Basic variables

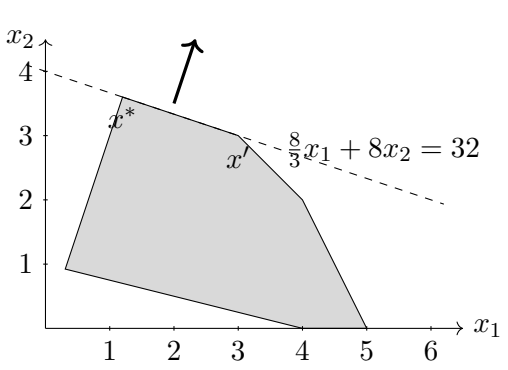
Consider once more the problem (7.1), which is represented in the diagram below.



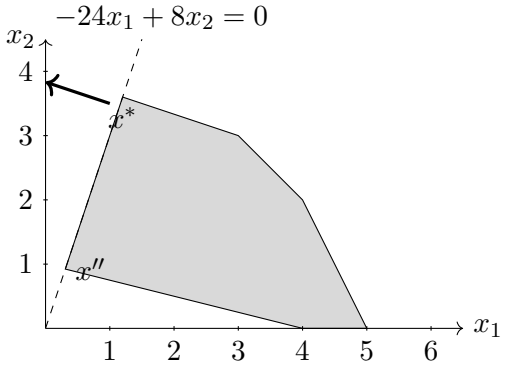
At the optimal solution x^*, of coordinates $x_1 = 1.2$, $x_2 = 3.6$, both variables are basic. Suppose we change the coefficient of x_1 in the objective function (which is $2x_1 + 8x_2$). As the

coefficient c_1 of x_1 increases, the optimal contour of the objective function rotates clockwise around the point x^*, until it becomes parallel to constraint 4 (which is $x_1 + 3x_2 \leq 12$). This happens when $c_1 = \frac{8}{3}$. At this value of c_1, the solution x^* is multiply optimal, and for $c_1 > \frac{8}{3}$ the optimal solution becomes the point x' at the intersection of constraints 3 and 4. (This situation is represented in Figure 7.3(a).)

Similarly, as the coefficient of x_1 decreases, the optimal contour of the objective function rotates counter-clockwise around the point x^*, until it becomes parallel to constraint 5 (which is $-3x_1+x_2 \leq 0$). This happens when $c_1 = -24$. At this value of c_1, the solution x^* is multiply optimal, and for $c_1 < -24$ the optimal solution becomes the point x'' at the intersection of constraints 5 and 6. (This situation is represented in Figure 7.3(b).)



(a) When c_1 is increased to $\frac{8}{3}$, the optimal extreme points are x^* and x'.



(b) When c_1 is decreased to -24, the optimal extreme points are x^* and x''.

Therefore the solution x^* remains optimal for all values of c_1 in the range

$$-24 \leq c_1 \leq \frac{8}{3}.$$

Note. In Excel Solver's sensitivity report, for every basic variable, the possible increase and decrease in the value of a coefficient for which the solution remains optimal are displayed as the allowable increase/decrease.

7.3 Example

Consider the laptop production example given in Section 2.1:

$$\begin{array}{lrcl}
\text{maximise} & 2x_1+1.4x_2+x_3+0.8x_4+0.5x_5 & & \\
\text{subject to} & x_1+x_2+x_3+x_4+x_5 \leq 8 & & \text{(CPU available)} \\
& x_1+0.7x_2+0.3x_4 \leq 3 & & \text{(SSDs available)} \\
& 6x_1+4x_2+4x_3+2x_4+2x_5 \leq 14 & & \text{(Memory boards availale)} \\
& x_1+x_2 \leq 5 & & \text{(Expected demand L1, L2)} \\
& x_3+x_4+x_5 \leq 4 & & \text{(Expected demand L3, L4, L5)} \\
& x_2 \geq 0.7 & & \text{(Orders for L2)} \\
& x_5 \geq 0.7 & & \text{(Orders for L5)} \\
& x_1,\, x_2,\, x_3,\, x_4,\, x_5 \geq 0. & &
\end{array}$$

The optimal solution given by the LP solver is the following:

Variable	Value	Basic	Allowable Increase	Allowable Decrease
x_1	0	no	0.1	∞
x_2	1.5	yes	0.2	0.0667
x_3	0	no	0.5	∞
x_4	3.3	yes	∞	0.1
x_5	0.7	yes	0.3	∞

The total profit is 5.09 million pounds. The dual solution is the following:

Constraint	Dual Value	Effective	Allowable Increase	Allowable Decrease
1	0	no	$+\infty$	2.5
2	0	no	$+\infty$	0.96
3	0.35	yes	5.4857	3.2
4	0	no	$+\infty$	3.5
5	0.1	yes	1.6	3.3
6	0	no	0.8	∞
7	-0.3	yes	3.3	0.7

The table also reports the allowable increases and decreases for the sensitivity analysis. The company is evaluating the following scenarios:

- *The company could purchase 2000 extra memory boards from a different supplier, at a cost of £200,000. Should they consider it?*
  The right-hand-side of constraint 3 would go from 14 to 16, which is within the range. Since the dual value of constraint 3 is 0.35, this will cause an increase of $2 \cdot 0.35 = 0.7$ in the optimal objective value. Therefore the company would increase its profits by £700,000, at a cost of £200,000. The company should purchase the extra boards.

- *Marketing estimates that spending £150,000 in advertising would boost demand for the lower priced laptops L3, L4, L5 by one thousand units in the next quarter. Should the company invest the money in advertising?*
  An increase in demand of 1000 units corresponds to an increase of 1 in the right-hand-side of the 5th constraint, from 4 to 5, which is within the range. Since the corresponding dual value is 0.1, the profit from sales would increase by £100,000, which does not justify spending £150,000 in advertising.

- *The company realises that it is loosing money on its cheapest line of laptops, so they intend to scrap production. However, the company would face a £120,000 penalty for the missed delivery of the orders that have already been placed. What should they do?*
  The right-hand-side of constraint 7 would go from 0.7 to 0, within the range. The company would save $(-0.7) \cdot (-0.3) = 0.21$ million pounds. This offsets the penalty of 0.12 million pounds.

- *The company realises that it has priced its top-of-the range laptop too low. At how much should they price it in order for it to become profitable?*
  The upper limit for the objective coefficient of variable x_1 is 2.1. This means that it will become profitable to produce laptops of type 1 if they are priced at most £2100. If they price them at more than that, they will need to resolve the problem to decide the new production plan.

- *Higher labour prices in the factory producing laptops of type 4 will reduce profit on each unit by £100. Should the company consider changing its production plan? How will this affect its profits?*
  The coefficient of variable x_4 in the objective will go from 0.8 to 0.7, within the range. This means that the optimal production plan will not change. The profits will, however, be reduced by $0.1 \times 3.3 = 0.33$ million pounds.

7.4 Exercises

Exercise 7.1. Consider the following LP:

$$\begin{array}{lrcl} \text{maximise} & x_1 + 9x_2 + 3x_3 & & \\ \text{subject to} & x_1 + x_2 + 0.5x_3 & \leq & 4 \\ & 5x_1 - 3x_2 - x_3 & \geq & 0 \\ & x_1, x_2, x_3 & \geq & 0. \end{array}$$

(a) Write the dual of the above LP, and draw a diagram of the dual (the dual has only two variables).

(b) Determine the optimal primal and dual solution, by finding the optimal dual solution from the diagram and proving optimality.

(c) By how much does the objective function coefficient of the nonbasic variables in the primal solution have to change to make the solution multiply optimal?

(d) How much does the right-hand-side of the second constraint in the primal have to decrease before the second constraint becomes ineffective in an optimal solution?

Exercise 7.2. Consider the following LP:

$$\begin{array}{lrcl} \text{minimise} & 4x_1 + x_2 + 1.5x_3 & & \\ \text{subject to} & 2x_1 + 8x_2 - 4x_3 + 4x_4 & = & 4 \\ & 3x_1 + 2x_2 + x_3 + 0.5x_4 & \geq & 2 \\ & -6x_1 - x_2 + 5x_3 + 2x_4 & \leq & 6 \\ & x_1, x_2, x_3, x_4 & \geq & 0. \end{array}$$

A proposed solution to this problem is $x = (0, 0.75, 0.5, 0)$. Show that this solution is feasible and that it is optimal. Is it a unique optimal solution? By how much can the following elements vary and the solution remains feasible and optimal?

(a) the right-hand-side of the ineffective constraints,

(b) the objective function elements of the nonbasic variables.

Exercise 7.3. A factory processes two products, P_1 and P_2, on two machines, M_a and M_b. The cost of raw materials is £250 for P_1 and £350 for P_2. The processing time and cost per lot of product on the two machines is given in the table below.

	P_1	P_2
M_a	3 hours £500	8 hours £1000
M_b	5 hours £450	5 hours £1200

The sale price per lot of product is £900 and £1800 for P_1 and P_2, respectively. Each machine is available 40 hours per week, and a weekly demand of 7 lots of product P_1 must be met. Write an LP model to determine the number of lots of P_1 and P_2 to be produced in each week in order to maximise the total profit. Solve the problem with Excel Solver.

(a) By how much should the price of P_1 increase before it becomes profitable to produce more than the demand?

(b) Suppose that, instead of producing P_1 at all, the factory could purchase from a third party the entire demand of 7 lots of P_1 for a price of £910 per lot. Would this be more profitable?

(c) By how much would the cost of producing P_1 on machine M_b have to decrease in order in order for M_b to start producing P_1 in an optimal plan?

(d) How much would the factory be willing to spend for each extra hour of operation of machine M_a?

(e) The factory has the option of producing a third product, P_3. This product can only be processed by machine M_a, requiring 2 hours and costing £550 per lot. The cost of raw material for P_3 is £250 per lot. Furthermore, product P_3 can be sold as an alternative to P_1 (i.e., the demand of 7 lots can be fulfilled using P_1 or P_3, indifferently). What is the minimum price sale price to be assigned to P_3, in order for it to be profitable?

Chapter 8

The Simplex method

Linear programming emerged after the end of World War II as a practical, powerful tool in a wide array of applications. This was made possible by the convergence of two events: the advent of computers, and the development of the first effective method for solving LP problems, the Simplex method. Devised by George Dantzig in the 1940s, the method remains today at the core of all commercial Linear Programming solvers. Solvers such as IBM's Cplex, Gurobi, or FICO's Xpress, to name a few, all implement some version of the method.

8.1 An example

Consider the following LP problem:

$$\begin{array}{rrcrcr} \text{maximise} & 3x_1 & + & 2x_2 & & \\ \text{subject to} & x_1 & + & x_2 & \leq & 6 \\ & x_1 & + & 2x_2 & \leq & 10 \\ & x_1 & - & x_2 & \leq & 4 \\ & & & x_1, x_2 & \geq & 0. \end{array}$$

Let us first transform the problem, so that all resource constraints are of type "=". This can be done by introducing some new nonnegative variables into the problem. For example, if we introduce a new variable x_3, and set $x_3 = 6 - x_1 - x_2$, then the first constraint is equivalent to the two constraints

$$\begin{array}{c} x_1 + x_2 + x_3 = 6 \\ x_3 \geq 0 \end{array}$$

Variable x_3 is often referred to in the literature as *slack variable* relative to the first constraints[1]. By introducing a slack variable x_4 for the second constraint, and a slack variable

[1]The name is justified because x_3 takes value equal to the "slack" of the corresponding constraint, that is, the difference between the right-hand-side and the left-hand-side of the constraint.

x_5 for the third, we can write the problem as:

$$\begin{array}{rrcrcrcrcrcr} \text{maximise} & 3x_1 & + & 2x_2 & & & & & & & & \\ \text{subject to} & x_1 & + & x_2 & + & x_3 & & & & & = & 6 \\ & x_1 & + & 2x_2 & & & + & x_4 & & & = & 10 \\ & x_1 & - & x_2 & & & & & + & x_5 & = & 4 \\ & & & & & & & x_1, x_2, x_3, x_4, x_5 & & & \geq & 0. \end{array}$$

Note that there is an obvious feasible solution to start from, namely the point $x^* = (0, 0, 6, 10, 4)$. Since all three resource constraints are effective, and there are three variables with nonzero value, i.e. basic, the solution is extreme. The value of the objective function at x^* is 0.

First iteration Can we find a better solution? Note that if we increase x_2 by $t \geq 0$ and leave $x_1 = 0$, the objective value increases by $2t$. However, to ensure that the resource constraints are satisfied, the remaining components must become:

$$\begin{aligned} x_3(t) &= 6 - t \\ x_4(t) &= 10 - 2t \\ x_5(t) &= 4 + t. \end{aligned}$$

We can only increase t as long as the above three variables take nonnegative values. What is the maximum value of t we can choose? In order to have $x_3(t) \geq 0$, we need $t \leq 6$. For $x_4(t) \geq 0$, we need $t \leq 5$. Note instead that $x_5(t) \geq 0$ for all nonnegative values of t. Thus, the largest t we can choose is $t = 5$.

The new solution we obtain is therefore the point $(0, 5, 1, 0, 9)$. Note that there are still three effective constraints and three basic variables (namely, x_2, x_3, x_5). The solution is extreme. The value of the solution is $2 \cdot 5 = 10$.

Second iteration Can we find a yet better solution? What we did before was to increase the value of a nonbasic variable with positive coefficient in the objective function. Since x_1 is nonbasic at the current point, and has a positive coefficient in the objective function, one might be tempted to increase its value. However, to do so, we might have to decrease the value of x_2, and we do not know, a priori, if the next solution will be better or worse than the current one.

What property did we use when we decided to increase x_2, that guaranteed that the next solution would be at least as good as the starting one (and, in fact, strictly better provided that $t > 0$)?

We used the fact that all basic variables had 0 coefficient in the objective function. When this occurs, if we increase the value of a nonbasic variable with positive objective coefficient, leaving at 0 the remaining nonbasic variable, and we adjust the values of the basic variables to maintain feasibility, we are guaranteed that the new solution we find will have better objective value, because the changes in the basic variables do not affect the value in the objective function.

So we would like only the nonbasic variables x_1, x_4 to appear in the objective function. To accomplish this, we resort to the following trick. Let us introduce a new variable z, and set

$$z = 3x_1 + 2x_2.$$

Now, the original problem can be stated as

$$\begin{array}{rrcrcrcrcrcrl}
\text{maximise} & z & & & & & & & & & & & \\
\text{subject to} & 3x_1 & + & 2x_2 & - & z & & & & & & = & 0 \\
& x_1 & + & x_2 & + & x_3 & & & & & & = & 6 \\
& x_1 & + & 2x_2 & & & + & x_4 & & & & = & 10 \\
& x_1 & - & x_2 & & & & & + & x_5 & & = & 4 \\
& & & & & & & x_1, x_2, x_3, x_4, x_5 & & & & \geq & 0.
\end{array}$$

Note that the second constraint of the original problem expresses the basic variable x_2 in terms of the nonbasic variables x_1, x_4, that is

$$x_2 = 5 - \frac{1}{2}x_1 - \frac{1}{2}x_4.$$

Substituting x_2 with $5 - \frac{1}{2}x_1 - \frac{1}{2}x_4$ in the equation defining z, we obtain

$$z = 10 + 2x_1 - x_4.$$

Therefore, if we increase the value of x_1 from 0 to t, while leaving the value of x_4 at 0, the next solution will have value $10 + 2t$, which is greater than the current one. We need to compute the values of the basic variables in order to ensure that the next solution satisfies the resource constraints. In order to do this, we need to know how x_2, x_3, x_5 change as x_1 increases. Thus, we need to solve the basic variables in terms of the nonbasic one. This can be done by substituting x_2 with $5 - \frac{1}{2}x_1 - \frac{1}{2}x_4$ in all the constraints of the problem.

We obtain the following problem, equivalent to the previous one:

$$\begin{array}{rrcrcrcrcrcrl}
\text{maximise} & z & & & & & & & & & & & \\
\text{subject to} & -2x_1 & & & & & + & x_4 & + & z & = & 10 \\
& \frac{1}{2}x_1 & & & + & x_3 & - & \frac{1}{2}x_4 & & & = & 1 \\
& \frac{1}{2}x_1 & + & x_2 & & & + & \frac{1}{2}x_4 & & & = & 5 \\
& \frac{3}{2}x_1 & & & & & + & \frac{1}{2}x_4 & + & x_5 & = & 9 \\
& & & & & & & x_1, x_2, x_3, x_4, x_5 & & & \geq & 0.
\end{array}$$

If we increase x_1 by $t \geq 0$ and leave $x_4 = 0$, the objective value z increases by $2t$.

The remaining components must become:s

$$\begin{aligned}
x_2(t) &= 5 - \frac{1}{2}t \\
x_3(t) &= 1 - \frac{1}{2}t \\
x_5(t) &= 9 - \frac{3}{2}t.
\end{aligned}$$

The variables x_2, x_3, x_5 remain nonnegative as long as t satisfies, respectively, $t \leq 10$, $t \leq 2$, and $t \leq 6$. Thus, the largest t for which the solution is nonnegative is $t = 2$. The new solution is $(2, 4, 0, 0, 6)$, with value 14. The variables x_1, x_2, and x_5 are basic, while x_3 and x_4 are nonbasic. Note that the variable x_1, which was nonbasic, has now become basic, while the variable x_3, which was basic, has now become nonbasic.

Third iteration As before, we want to express z and the basic variables x_1, x_2, x_5 in terms of the nonbasic variables x_3, x_4. We can use the constraint $\frac{1}{2}x_1 + x_3 - \frac{1}{2}x_4 = 1$ to express x_1 in terms of x_3 and x_4, namely

$$x_1 = 2 - 2x_3 + x_4.$$

Substituting x_1 with $2 - 2x_3 + x_4$ in all constraints of the previous LP, we get

$$\begin{array}{rrrrrrrrrrrrr}
\text{maximise} & z & & & & & & & & & & & \\
\text{subject to} & & & & & + & 4x_3 & - & x_4 & + & z & = & 14 \\
& x_1 & & & & + & 2x_3 & - & x_4 & & & = & 2 \\
& & + & x_2 & & - & x_3 & + & x_4 & & & = & 4 \\
& & & & & - & 3x_3 & + & 2x_4 & + & x_5 & = & 6 \\
& & & & & & & & x_1, x_2, x_3, x_4, x_5 & & & \geq & 0.
\end{array}$$

If we increase x_4 by $t \geq 0$ and leave $x_3 = 0$, the objective value increases by t. The remaining components must become

$$\begin{aligned}
x_1(t) &= 2 + t \\
x_2(t) &= 4 - t \\
x_5(t) &= 6 - 2t.
\end{aligned}$$

The largest value of t for which the basic variables are nonnegative is $t = 3$. The new basic solution is $(5, 1, 0, 3, 0)$, with value 17.

Fourth iteration Substituting x_4 with $3 + \frac{3}{2}x_3 - x_5$ in all constraints, we obtain:

$$\begin{array}{rrrrrrrrrrrrr}
\text{maximise} & z & & & & & & & & & & & \\
\text{subject to} & & & & & \frac{5}{2}x_3 & & & + & \frac{1}{2}x_5 & + & z & = & 17 \\
& x_1 & & & + & \frac{1}{2}x_3 & & & + & \frac{1}{2}x_5 & & & = & 5 \\
& & + & x_2 & + & \frac{1}{2}x_3 & & & - & \frac{1}{2}x_5 & & & = & 1 \\
& & & & - & \frac{3}{2}x_3 & + & x_4 & + & \frac{1}{2}x_5 & & & = & 3 \\
& & & & & & & & x_1, x_2, x_3, x_4, x_5 & & & \geq & 0.
\end{array}$$

Can there be a better solution? Since the coefficients of x_3 and x_5 in the objective function are negative, it is impossible to achieve a value better than 17. Therefore our current solution is optimal.

We have visited the following solutions:

$$(0, 0, 6, 10, 4) \to (0, 5, 1, 0, 9) \to (2, 4, 0, 0, 6) \to (5, 1, 0, 3, 0).$$

The components relative to the original variables x_1 and x_2 can be plotted in a diagram, shown in Figure 8.1.

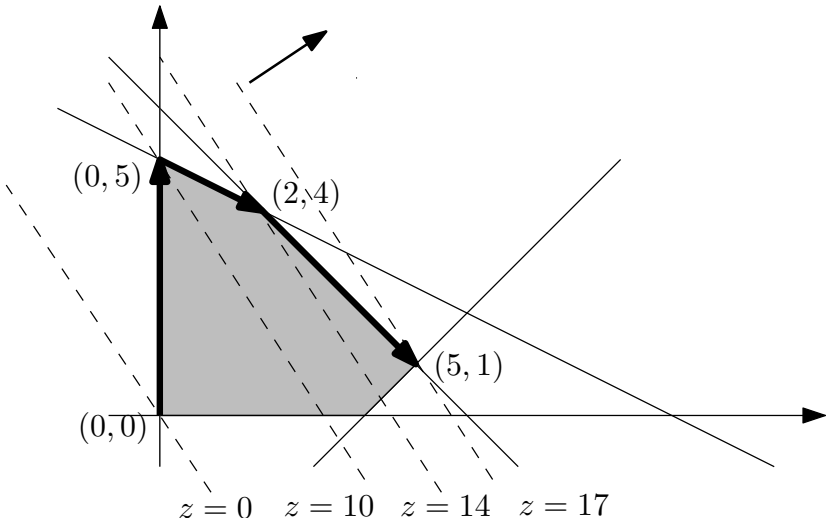


Figure 8.1: Sequence of points visited by the Simplex method.

8.2 Description of the Simplex method

In order to be applied, the Simplex method requires the problem to be converted to an equivalent one in a form where all variables are nonnegative and where all resource constraints are of type "=". This can always be done by introducing new variables (as we did in the previous example). Note that LP solvers accept LP problems in any form, so the user need not be concerned with entering the problem in the appropriate form.

Thus, we assume that we are dealing with an LP problem in the form

$$\begin{array}{rrcl} \text{maximise} & c^\top x & & \\ \text{subject to} & Ax & = & b \\ & x & \geq & 0, \end{array} \tag{8.1}$$

where A is an $m \times n$ matrix, c is an n-dimensional row vector, b is an m-dimensional column vector, and x is a column vector of n variables.

We will also assume that the m resource constraints are independent. Since all resource constraints are equalities and they are independent, every basic solution will have m basic variables and $n - m$ nonbasic variables.

It will be convenient to write (8.1) in the following equivalent form:

$$\begin{array}{rrcl} \text{maximise} & z & & \\ \text{subject to} & c^\top x - z & = & 0 \\ & Ax & = & b \\ & x & \geq & 0 \end{array} \tag{8.2}$$

The Simplex method needs to be given an initial extreme solution from which to start, say $\bar{x}$, and the corresponding set of basic variables. (In the example in the previous section, the initial extreme solution was the point $(0, 0, 6, 10, 4)$.)

Let us assume that the basic variables are $x_{B[1]}, x_{B[2]} \ldots, x_{B[m]}$, and let us denote by N the set of indices of the $n-m$ nonbasic variables. In the previous example, for the extreme point $(0,0,6,10,4)$, we had $x_{B[1]} = x_3$, $x_{B[2]} = x_4$, $x_{B[3]} = x_5$, and $N = \{1,2\}$.

We solve the variables z, $x_{B[1]}, x_{B[2]} \ldots, x_{B[m]}$ in terms of the nonbasic variables, so that the problem is of the form

$$\begin{array}{lrcl} \text{maximise} & z & & \\ \text{subject to} & -z + \sum_{j\in N} \bar{c}_j x_j & = & 0 \\ & x_{B[i]} + \sum_{j\in N} \bar{a}_{ij} x_j & = & \bar{b}_i, \quad i = 1,\ldots,m \\ & x & \geq & 0. \end{array} \tag{8.3}$$

The extreme solution $\bar{x}$ is defined by

$$\begin{aligned} \bar{x}_{B[i]} &= \bar{b}_i, \quad i = 1,\ldots,m \\ \bar{x}_j &= 0, \quad j \in N \end{aligned}$$

(so in particular $\bar{b}_1, \ldots, \bar{b}_m \geq 0$), and has objective value $\bar{z}$.

There are two possible cases:

Case 1. There exists a nonbasic variable x_k such that $\bar{c}_k > 0$.

Ideally, we would like to find a new feasible basis whose objective value is better (or not worse) than $\bar{z}$. Note that, since $\bar{c}_k > 0$, if we increase the value of x_k from 0 to some number $t \geq 0$, while leaving at 0 the value of all other nonbasic variables, then the value of the new solution in the objective function will increase by $\bar{c}_j t$, thus improving the value of the objective function. However, in order to do so, we need to adjust the values of the basic variables to maintain feasibility.

More formally, for $t \geq 0$, the basic variables need to be defined as

$$x_{B[i]}(t) = \bar{b}_i - t\bar{a}_{ik}, \qquad i = 1,\ldots,m. \tag{8.4}$$

By construction, the new solution satisfies the resource constraints. We would like to increase the value of t as much as possible. The only thing preventing t from increasing indefinitely is the nonnegativity of the variables. Thus we can have two subcases:

a) For some value of $t = t^*$, the value of some basic variable, say x_ℓ, goes to 0, while keeping the values of all other variables nonnegative. In this case, the new extreme solution is obtained from (8.4) for $t = t^*$. The variable x_k becomes basic, while the variable x_ℓ become nonbasic, and we repeat.
(In the first iteration in the example in the previous section, we had $x_k = x_2$, $x_\ell = x_4$, $t^* = 5$, and the new solution was $(0, t^*, 6 - t^*, 0, 4 + t^*)$.)

b) The values of the variables remain nonnegative for any positive value of t.
In this case we can increase the value of t as much as we want, thus obtaining feasible solutions of arbitrarily large value. It follows that the problem is unbounded.

Case 2. $\bar{c}_j \leq 0$ for all indices $j \in N$.

This means that all coefficients in the objective function of the LP (8.3) are ≤ 0. So no solution can have value greater than $\bar{z}$. Since $\bar{x}$ has value $\bar{z}$, it must be optimal. Thus the algorithm stops.

The Simplex method

Start from an extreme solution, with basic variables $x_{B[1]}, \ldots, x_{B[m]}$.

1. Write the LP in the form (8.3) by expressing the variables $z, x_{B[1]}, \ldots, x_{B[m]}$ in terms of the nonbasic variables.
2. If $\bar{c}_j \leq 0$ for all $j \in N$, then the current solution is optimal. STOP.
3. Otherwise, pick a nonbasic variable x_k such that $\bar{c}_k > 0$.
   Compute the largest value t^* of t such that the solution (8.4) is feasible.
   - 3a. If $t^* < +\infty$, then some basic variable x_ℓ takes value 0. Compute the new extreme solution, and replace x_ℓ with x_k as a basic variable. Return to 1.
   - 3b. If $t^* = +\infty$, then the problem is unbounded. STOP.

8.3 Exercises

Exercise 8.1. Consider the LP problem $\max\{c^\top x \mid Ax = b,\ x \geq 0\}$, where

$$A = \begin{pmatrix} 1 & 2 & 0 & 1 & 0 & -6 \\ 0 & -1 & -1 & -3 & 2 & 1 \\ 1 & 2 & 1 & 3 & -1 & 5 \end{pmatrix}, \quad b = \begin{pmatrix} 11 \\ -6 \\ 13 \end{pmatrix}, \quad c = \begin{pmatrix} -2 \\ -2 \\ 3 \\ 4 \\ -10 \\ 5 \end{pmatrix}$$

(a) Show that $B = \{2, 4, 5\}$ is a feasible basis. Transform the problem to the form where the basic variables are expressed in terms of the nonbasic variables.

(b) Apply the Simplex method to solve the LP problem starting from the feasible basis $B = \{2, 4, 5\}$. At any iteration, if you have a choice between multiple nonbasic variables that can enter the basis, choose the one with the smallest possible index. (This is known as Bland's rule.)

Exercise 8.2. Consider the following linear program:

$$\begin{array}{lrcrcl} \max & 2x_1 & + & x_2 & & \\ & x_1 & - & 2x_2 & \leq & 2 \\ & 2x_1 & - & x_2 & \leq & 5 \\ & -4x_1 & + & x_2 & \leq & 4 \\ & x_1 & , & x_2 & \geq & 0. \end{array}$$

Transform the above problem in standard equality form by adding the slack variables x_3, x_4, x_5, relative to the first, second, and third constraint, respectively. Note that $\{3, 4, 5\}$ is a feasible basis, and apply the Simplex method starting from this basis.

The problem will be unbounded. From the final step of the Simplex algorithm, determine a family of feasible solutions that can take arbitrarily large objective value.

Chapter 9

Network problems

Directed graphs already proved to be very useful in modelling a variety of problems, such as shortest paths, project management, and Markov chains. In this chapter, we investigate Linear Programming formulations for problems in directed graphs.

9.1 Network flow problem

Consider an organisation that has to supply a commodity to a number of destinations from some sources. The commodities have to be transported from the sources to the destinations through a network whose routes have costs. Every destination can meet its demand from every source, or from a combination of multiple sources. We wish to meet demand at the destinations from the supply at the sources at minimum total cost. The total supply of the item is equal to the total quantity required. One example is represented in Figure 9.1. Vertices 1 and 2 are the sources, as shown by the incoming arrows, vertices 4 and 6 are the destinations, as shown by the outgoing arrows, and vertices 3 and 5 are intermediate vertices.

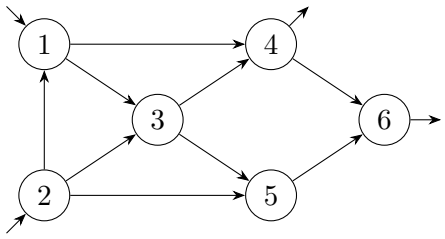


Figure 9.1: Representation of a network flow problem.

The cost of sending a unit of the commodity along the arcs are:

arc	(1,3)	(1,4)	(2,1)	(2,3)	(2,5)	(3,4)	(3,5)	(4,6)	(5,6)
cost	6	7	9	4	3	11	13	2	5

and available and required amounts are:

available	
vertex 1	90
vertex 2	55

demanded	
vertex 4	64
vertex 6	81

Note that the total demand and the total supply are the same, namely 145.

Decision variables We let x_{ij} denote the quantity of goods to be sent from vertex i to vertex j along arc (i, j). This will be referred to as the *flow* on arc (i, j).

Constraints Besides the nonnegativity of the variables, there are three types of constraints.

- *At a supply vertex*: the sum of the flow on the arcs going out of the vertex minus the sum of the flow on the arcs coming into the vertex should be at most the supply available at the vertex.
- *At a demand vertex*: the sum of the flow on the arcs coming into the vertex minus the sum of the flow on the arcs going out of the vertex should be at least the demand required at the vertex.
- *At an intermediate vertex*: the sum of the flow on the arcs coming into the vertex minus the sum of the flow on the arcs going out of the vertex should equal zero.

Note that, as the total availability of the item is equal to the total supply, all the above constraints are equalities. In the above example, we obtain the following constraints:

$$\begin{array}{lrcl}
\text{vertex 1:} & x_{21} - x_{13} - x_{14} & = & -90 \\
\text{vertex 2:} & -x_{21} - x_{23} - x_{25} & = & -55 \\
\text{vertex 3:} & x_{13} + x_{23} - x_{34} - x_{35} & = & 0 \\
\text{vertex 4:} & x_{14} + x_{34} - x_{46} & = & 64 \\
\text{vertex 5:} & x_{25} + x_{35} - x_{56} & = & 0 \\
\text{vertex 6:} & +x_{46} + x_{56} & = & 81.
\end{array}$$

Objective function

$$\text{minimise} \quad 6x_{13} + 7x_{14} + 9x_{21} + 4x_{23} + 3x_{25} + 11x_{34} + 13x_{35} + 2x_{46} + 5x_{56}.$$

Flow integrality property The constraints of the network flow problem ensure that, in general, whenever all demands and all available supplies are integer numbers, the extreme points have all integer components. So, for example, we are guaranteed that any optimal extreme solution of the above problem will assign integer valued flows to each arc.
This is particularly useful if the commodities we need to transport are indivisible (for example, cars or people), because in this case we would like the solution to be composed of only integer numbers. For general linear programming problems, finding the best solution were all variables take integer values can be an extremely challenging task.

9.2 The shortest path problem revisited

In Chapter 10 we studied the shortest path problem, and described Dijkstra's algorithm, an efficient method for finding a shortest path in a directed graph with nonnegative edge lengths. We now formulate this problem as a special case of the network flow problem in the previous section. We will then study the dual linear program. It turns out that the final distance estimates $d(u)$ in Dijkstra's algorithm can be interpreted as a dual optimal solution. We will illustrate the formulation with the example in Figure 9.2, the same example as in Chapter 10.

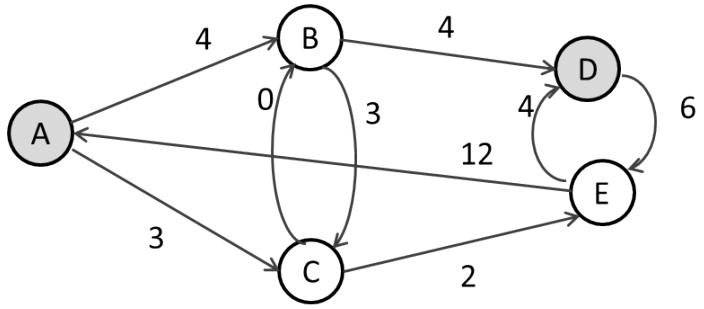


Figure 9.2: Example of a shortest path problem between vertices A and D.

The input to the shortest path problem is a directed graph $D = (V, A)$, with nonnegative edge lengths $\ell : A \to \mathbb{R}_+$, a starting vertex $s \in V$ and a target vertex $t \in V$. We define a network flow problem where one unit of flow has to be sent from the source s to destination t. That is, we set s to be the single source with one unit available, and t to be the single sink with one unit demanded. The length values ℓ are used as costs.

As noted above, this problem has an integer solution; moreover, it will have values 0 or 1 on every arc. The 1 arcs will form a path from s to t, and this will be in fact a shortest path. Later, we will see that Dijkstra's algorithm gives a proof of this fact.

Let us now formulate the LP for the example in Figure 9.2.

$$
\begin{array}{lrcl}
\text{minimise} & 4x_{AB} + 3x_{AC} + 4x_{BD} + 3x_{BC} + 0x_{CB} + 2x_{CE} + & & \\
& +6x_{DE} + 12x_{EA} + 4x_{ED} & & \\
\text{subject to} & x_{EA} - x_{AB} - x_{AC} & = & -1 \\
& x_{AB} + x_{CB} - x_{BD} - x_{BC} & = & 0 \\
& x_{AC} + x_{BC} - x_{CE} - x_{CB} & = & 0 \\
& x_{CE} + x_{DE} - x_{ED} - x_{EA} & = & 0 \\
& x_{BD} + x_{ED} - x_{DE} & = & 1 \\
& x_{AB}, x_{AC}, x_{BD}, x_{BC}, x_{CB}, x_{CE}, x_{DE}, x_{EA}, x_{ED} & \geq & 0
\end{array}
\tag{9.1}
$$

The first constraint stipulates that 1 unit of flow leaves A, and the last constraint stipulates that 1 unit of flow reaches D. The three other constraints enforce that for each of the other vertices, the total flow on the incoming arcs equals the total flow on the outgoing arcs.

The general form of the shortest path LP We can write the general problem in the following compact form. For every vertex $u \in V$, let $\delta^{in}(u)$ denote the set of all incoming arcs to u, and let $\delta^{out}(u)$ denote the set of all outgoing arcs from u. For an arc set $F \subseteq A$, we let $x(F) = \sum_{e \in F} x_e$.

Using this notation, the quantity $x(\delta^{in}(u)) - x(\delta^{out}(u))$ denotes the total flow on arcs incoming to the vertex u minus the total flow on arcs coming out from u. This has to equal 1 for $u = t$; has to equal -1 for $u = s$; and has to equal 0 in all other cases. The LP can be written as follows:

$$\begin{array}{lrcll} \text{minimise} & \sum_{e \in A} \ell(e) x_e & & & \\ \text{subject to} & x(\delta^{in}(s)) - x(\delta^{out}(s)) & = & -1 & \\ & x(\delta^{in}(u)) - x(\delta^{out}(u)) & = & 0 & \forall u \in V \setminus \{s,t\} \\ & x(\delta^{in}(t)) - x(\delta^{out}(t)) & = & 1 & \\ & x_e & \geq & 0 & \forall e \in A. \end{array} \tag{9.2}$$

In the example, we have $\delta^{in}(B) = \{(A,B),(C,B)\}$, $\delta^{out}(B) = \{(B,D),(B,C)\}$, $x(\delta^{in}(B)) = x_{AB} + x_{CB}$, and $x(\delta^{out}(B)) = x_{BD} + x_{BC}$. We see that the second constraint in (9.1) is the same as $x(\delta^{in}(B)) - x(\delta^{out}(B)) = 0$.

9.2.1 The dual problem

Let us start by writing the dual of the problem (9.1).

Variables The dual variables correspond to the constraints. Since every constraint corresponds to a vertex, we denote the dual variables by y_A, y_B, y_C, y_D, and y_E. Here, y_A corresponds to the first constraint, which stipulates that the incoming minus the outgoing flow at A should equal -1; y_B to the second constraint enforcing that the incoming minus the outgoing flow at B equals 0, etc. Since all primal constraints are equality constraints, all dual variables are free.

Constraints The dual constraints correspond to the primal variables. That is, there is a dual constraint corresponding to each arc of the graph. Since the primal is a minimisation problem and the variables are all nonnegative, the dual constraints will be $\leq$.

Consider the primal variable x_{AB} for the arc AB. This appears in two constraints: with coefficient -1 in the first constraint and with coefficient 1 in the second constraint. Thus, we have a dual constraint $-y_A + y_B \leq \ell(A,B)$.

Objective Since the primal problem was minimising, the dual will be maximising. There are only two nonzero entries in the right-hand-side of the primal problem: -1 for A and 1 for D. Hence, the objective is to maximise $y_D - y_A$.

We are ready to write the dual of (9.1):

$$\begin{array}{lllr} \text{maximise} & y_D - y_A & & \\ \text{subject to} & y_B - y_A & \leq & 4 \\ & y_C - y_A & \leq & 3 \\ & y_D - y_B & \leq & 4 \\ & y_C - y_B & \leq & 3 \\ & y_B - y_C & \leq & 0 \\ & y_E - y_C & \leq & 2 \\ & y_E - y_D & \leq & 6 \\ & y_A - y_E & \leq & 12 \\ & y_D - y_E & \leq & 4. \end{array} \tag{9.3}$$

The general form of the dual Let us now write the dual of the LP (9.2). The dual variables correspond to the vertices in V; we let y_u denote the dual corresponding to the vertex u. The objective is to maximise $y_t - y_s$.

We have a $\leq$ constraint for every arc (u, v). The variable x_{uv} occurs with coefficient 1 in the constraint for v (as it is an incoming arc at v), and with coefficient -1 in the constraint for u (as it is an outgoing arc at u). Thus, we can write the constraint as $y_v - y_u \leq \ell(u, v)$.

We obtain the dual of (9.2):

$$\begin{array}{llll} \text{maximise} & y_t - y_s & & \\ \text{subject to} & y_v - y_u & \leq \ \ell(u,v) & \forall (u,v) \in A. \end{array} \tag{9.4}$$

A simple transformation Let us observe that if we have a solution y to (9.4), then we can obtain another solution by adding the same real number λ to every y_u. That is, we can define $y'_u = y_u + \lambda$. Then y' will be another feasible solution with the same objective value.

This follows from the fact that $y'_v - y'_u = (y_v + \lambda) - (y_u + \lambda) = y_v - y_u$.

In particular, we can use $\lambda = -y_s$, thus setting $y'_s = 0$. Therefore, we can always find an optimal dual solution that has value 0 at the source s. This will be particularly convenient when making a connection to Dijkstra's algorithm.

9.2.2 Connection to Dijkstra's algorithm

Recall the distance labels $d(u)$ in Dijkstra's algorithm. At termination, these were the shortest path distances from s. Recall, in particular, that $d(s) = 0$, and for every arc $(u, v) \in A$, $d(v) \leq d(u) + \ell(u, v)$ must hold, with equality on the arcs of a shortest path. Therefore, $y_u = d(u)$ satisfies the inequalities of (9.4).

Optimality conditions Consider now the shortest path P returned by Dijkstra's algorithm. We can define $x_e = 1$ for every $e \in P$ and $x_e = 0$ for every $e \notin P$. This gives a feasible solution to (9.2). For the final distance labels, we set $y_u = d(u)$ and obtain a feasible solution to (9.4).

These solutions satisfy the optimality conditions in Section 5.3.2. Indeed, the basic variables correspond to the arcs in P, and for every arc $e = (u, v) \in P$, the dual inequality holds

with equality, that is, $y_v - y_u = \ell(u, v)$. This testifies that x is a primal optimal solution and y is a dual optimal solution.

Hence, Dijkstra's algorithm is an efficient method to solve the shortest path LP (9.2). Moreover, it is worth noting that we obtain an integer optimal solution.

We claimed that the network flow problem always has an integer optimal solution. Dijkstra's algorithm thus provides a proof of this statement for the special case of the shortest path LP.

Nonnegative arc lengths The LP formulation also reveals the importance of using nonnegative arc lengths. Let us modify the length $\ell(B, C)$ to -3 in Figure 9.2. This creates a negative length cycle consisting of the arcs (B, C) and (C, B). In this case (9.1) becomes *unbounded*. To see this, select an arbitrary large number M, and increase both x_{BC} and x_{CB} by M. This gives a feasible solution, and decreases the objective value by $3M$.

9.3 Transportation problem

Let us now study an other important special case of the network flow problem. In contrast to the shortest path problem, we allow multiple sources and destinations. On the other hand, we consider the situation where there are no intermediate vertices, that is, the sources are connected directly to the destinations. The graph of a model with two sources (S1 and S2) and three destinations (D1 D2 and D3) is shown in Figure 9.3.

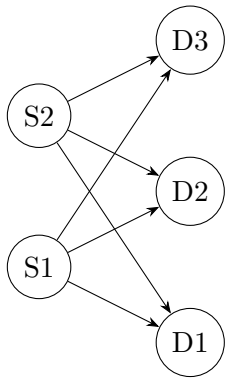


Figure 9.3: Representation of a transportation problem with two sources and three destinations.

In general, a transportation problem has the following structure:

- There are m sources, where each source $i = 1, \ldots, m$ has an available supply of a_i.
- There are n destinations, where each destination $j = 1, \ldots, n$ has a demand of b_j.
- For each source-destination pair i, j there is a unit transportation cost (or profit, depending on the context) c_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$.

- The total supply equals the total demand, that is

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j.$$

The objective is to minimise the total transportation cost (or maximise total transportation profit, depending on the context) in order to satisfy demand at the destination vertices.

Formulation Each variable x_{ij} represents the amount to be transported from source i to destination j, for $i = 1, \ldots, m$, $j = 1, \ldots, n$. All variables are nonnegative. Constraints of this problem can be written as in the more general network flow problem. The formulation is

$$\begin{array}{rrcll} \text{minimise (or maximise)} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} & & & \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} -x_{ij} & = & -a_i & i = 1, \ldots, m \\ & \displaystyle\sum_{i=1}^{m} x_{ij} & = & b_j & j = 1, \ldots, n \\ & x_{ij} & \geq & 0 & i = 1, \ldots, m,\ j = 1, \ldots, n. \end{array} \tag{9.5}$$

Observation Note that this formulation assumes that the total amount available is equal to the total amount required. However, this is not a very stringent condition since if there is more available than required we can invent a dummy $(n+1)^{\text{th}}$ destination to receive the surplus $b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$ at zero cost from each source, $c_{i,n+1} = 0$.

If there is more required than available, we can similarly invent an additional dummy $(m+1)^{\text{th}}$ source with $a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$ and with costs to each destination determined by the problem context (for example, it could be the unit cost of not meeting demand at each destination).

9.3.1 An application: targeted Internet advertising

Goble is a large company hosting Internet sites, and it derives a large part of its revenue from advertising. Its customers are other companies intending to advertise their services and products on Goble's webpages. When an ad is displayed on a page, Goble is paid a fee if the visitor clicks on the link.

To increase yield, Goble intends to resort to targeted advertising. For example, visitors on a sports news page may be more inclined to click on an advertisement of sporting goods than, say, readers of a fashion forum.

Webpages hosted by Goble are divided into context clusters, such as, sports, entertainment, weather, politics, etc. Let m be the number of clusters. In a given time unit, let us say that there are n advertisements to be displayed.

- Goble has an estimate of the probability visitors will click on advertisement j when this is displayed on a page in cluster i; say p_{ij} ($i = 1, \ldots, m$, $j = 1, \ldots, n$).
- Goble's customers want their ad to appear in at least a certain number of pages in the given time unit; let b_j be the minimum number of times that ad j should appear, $j = 1, \ldots, n$.
- In a given time unit, only a limited number of adds can appear in each cluster; let a_i be the maximum number of adds that can appear in cluster i in the given time unit, $i = 1, \ldots, m$.

Goble's objective is to maximise the expected total number of clicks on the advertisements displayed.

The problem can be cast as a (maximisation) transportation problem as follows:

- Each cluster i corresponds to a supply vertex, with available supply a_i;
- Each ad j corresponds to a demand vertex, with demand b_j;
- The transport "profit" from source i to destination j is the p_{ij}.

For each ad $i = 1, \ldots, n$ and each cluster $j = 1, \ldots, m$, the variable x_{ij} now represents the number of times ad j appears in cluster i. The objective function is to maximise total profit, that is

$$\text{maximise} \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} x_{ij},$$

which is to maximise the total expected number of times that visitors will click on some ad.

9.4 Exercises

Exercise 9.1. Formulate the dual of the transportation linear program (9.5).

Exercise 9.2. A firm can use 3 different methods to make a product. All the methods use machines A, B, and C. For example, making one unit of the product by Method 1 requires 2 hours on Machine A, 4 hours on Machine B, and 3 hours on Machine C. The table below gives the hours of machine time required by each method.

	Method		
	1	2	3
Machine A	2	1	3
Machine B	4	2	3
Machine C	3	4	2

50 hours are available on each machine. The profit from one unit of the product depends on the method used:

Method	1	2	3
Profit (£)	15	18	10

(a) Formulate this problem so as to maximise profit.

(b) Formulate this problem so as to minimise the number of hours during which Machine B is used, subject to profit being at least £200.

Exercise 9.3. A company operates by buying a commodity during some months and selling it during some other months. The market price for selling or for buying is (per tonne):

Month	January	February	March	April
Price (£)	60	90	80	110

Storing costs £10 per tonne per month, and the store cannot take more than 20 tonnes at any one time. Buying and selling take place at the beginning of the month. There is no stock at the beginning of January, and no stock is required at the end of April.

Formulate this problem, and solve it via Excel Solver. Confirm that the optimal policy is to buy and sell in no months except:

Month	action	amount
January	buy	20
February	sell	20
March	buy	20
April	sell	20.

Exercise 9.4. A company has 3 factories and 4 shops selling its product. Each week the product is transported from the factories to the shops. The 3 factories can produce 50, 70, and 60 tonnes respectively in a week. The 4 shops sell 25, 65, 45, and 35 tonnes respectively a week. The costs of transporting a tonne of product from factories to shops is given in the table below (in £s per tonne):

	Shop 1	Shop 2	Shop 3	Shop 4
Factory 1	20	18	8	12
Factory 2	7	12	4	16
Factory 3	10	6	15	10

Formulate the company's problem, and solve it using Excel Solver. What would be your formulation if:

(a) the third factory produced only 30 units?

(b) the second shop required only 30 units?

Chapter 10

The Shortest Path problem

Finding a shortest path in a network is a fundamental optimisation problem. An everyday application is using our smartphones for planning routes for getting around. Path length can be measured in various ways, notably in distance or travel time, and shortest routes for different measures may be different: using motorways may result in longer distance but shorter travel time. Further, we can obtain fundamentally different optimal routes when planning for driving, public transport, and walking: in these cases, the underlying networks are different, even if they all represent the same city. For example, small footpaths or stairs can only be used only in walking routes, while tube tunnels can be used only in public transportation routes.

The *shortest path problem* also appears in non-geographical contexts. Routing data through the Internet relies on finding shortest paths subject to various requirements. Or, if you wish to reach through personal contacts someone you do not already know, you may want to find a shortest path in your social network: one going through a smallest-possible number of intermediate people. The famous (if not fully justified) "six degrees of separation" theory claims that every person on Earth can reach every other person through a friends-of-friends chain of length at most six.

The typesetting system that created these notes (LaTeX) considers many possible ways of breaking a paragraph into lines. Each possible line, given by a starting point (a word or syllable) and an ending point, is assigned a "badness" value depending on how far it is from an ideal length. The best formatting of the paragraph is the one with minimum total badness, and is given by the shortest path from the paragraph's first word to its last, through some among the intermediate points (line breaks).

Directed graphs are a standard and convenient mathematical model for all these networks.

A *directed graph* $G = (V, A)$ comprises a set of *vertices* V and a set A of ordered pairs of vertices called *arcs*.

We will often omit "directed" and simply refer to graphs. An arc $(u, v) \in A$ is thought of as a directed edge from vertex $u \in V$ to vertex $v \in V$. The *out-neighbours* of a vertex u are all vertices v with $(u, v) \in A$, and the *in-neighbours* are all vertices v with $(v, u) \in A$. Figure 10.1 shows a standard representation of a graph, with arrows representing the arcs.

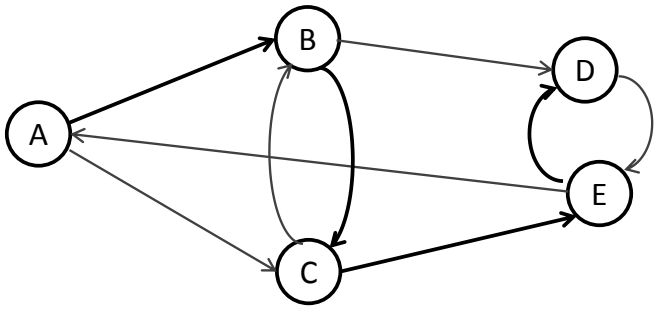


Figure 10.1: Example of a directed graph, with the bold arcs representing the path *A–B–C–E–D*.

In a driving route planning problem for London, vertices would correspond to junctions, and arcs to road segments between two junctions. To be more precise, arcs do not represent roads but roads with direction: the arcs in our model are directed. This enables modelling of one-way roads, and also makes it possible to model situations where the length in the two directions can be different. E.g., if "length" represents traversal time, it can be quite different uphill and downhill, and also depending on the traffic situation (rush hour traffic will be slow in one direction and fast in the other). If a road segment can be used in both directions, we represent it by a pair of oppositely directed arcs. Note that in the representation in Figure 10.1, the relative positions of the vertices do not matter at all: the arcs capture all the relevant information. In the social network model, vertices correspond to individuals, and arcs can represent a type of connection. For example, an arc pointing from u to v may express that u knows v: this is an asymmetric relation, with famous people having many more arcs pointing into them than going out from them. On the other hand, friendship is a symmetric relation, better represented by an undirected graph. (In undirected graphs, the connections are usually called *edges* rather than arcs.)

We are interested in finding shortest paths in a graph, so we formally define a path.

> In a directed graph $G = (V, A)$, a *directed path* from a vertex s to another vertex t is a sequence of vertices $u_0, u_1, \ldots, u_k$ with $u_0 = s, u_k = t$, such that (u_i, u_{i+1}) is an arc for every $i = 0, \ldots, k-1$, and the same vertex cannot appear twice in the sequence.

Figure 10.1 shows a directed path from vertex A to vertex D. There are altogether 4 directed paths from A to D: *find them all.*

The *cardinality of a path* is the number of arcs contained in the path. For example, the path ABD in Figure 10.1 has cardinality 2, whereas ABCED has cardinality 4.

We will sometimes omit the adjective "directed" and just say "paths". The notation $u_0u_1 \ldots u_k$ will be also used (omitting the commas). We will often treat a path as a sequence of arcs (as opposed to a sequence of vertices). We are now ready to formally define the problem.

We are given a directed graph $G = (V, A)$, a length function $\ell : A \to \mathbb{R}$ on the arcs, and two vertices $s, t \in V$. The *shortest path problem* is to find a directed path between s and t that minimises the length of the path.

The length of a path is by definition the sum of the lengths of its edges, so a path $P = u_0 u_1 \dots u_k$ has length

$$\ell(P) = \sum_{i=0}^{k-1} \ell(u_i, u_{i+1}).$$

Note that there may be no path from s to t whatsoever; for simplicity, we will ignore this uninteresting possibility. Figure 10.2 extends the previous example by adding lengths. Of all the directed A—D paths, the shortest one is A–C–B–D, of length 7.

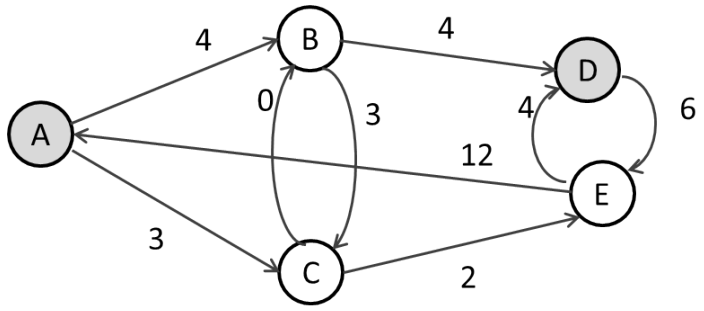


Figure 10.2: Example of a shortest path problem between vertices A and D.

Let us also define d to be the distance between any two points, i.e., the length of the shortest path between them:

$$d(s, t) = \min \{\ell(P) \colon P = s, \dots, t\}.$$

10.1 Nonnegative arc lengths: Dijkstra's algorithm

In the definition of a shortest path, the length function $\ell : A \to \mathbb{R}$ is allowed to take arbitrary real values, and we shall see in later sections that using *negative* values is essential in certain applications. However, in all the examples above, it is natural to assume that ℓ is nonnegative: if ℓ represents distance or time duration, negative values do not make sense. For nonnegative cost functions, a shortest path can be efficiently computed by a simple and natural algorithm, described by Edsger W. Dijkstra in 1959.[1]

[1]The term *"algorithm"* will not be formally defined here. Loosely speaking, it is a well-described step-by-step method to achieve a given task, and can be implemented in a computer. We also do not define what an "efficient algorithm" means: it is one that is guaranteed to finish in a small number of steps, but then we must say exactly what we mean by both "small" and "step". The interested reader may refer to [Pa] for an introduction to the field of complexity of algorithms.

10.1.1 Idea of the algorithm

We start with a high level overview. The algorithm will find in order the vertex closest to the source vertex s, second closest, and so on, with the corresponding distances. We can stop the algorithm when a given target vertex t is reached, or we can simply run it until all the distances to all vertices have been computed. Since the source vertex s is given, we will abbreviate distances $d(s, v)$ simply as $d(v)$. We will always start with $v_1 = s$, at distance $d(s) = d(s, s) = 0$. This relies on the hypothesis that all distances are nonnegative: otherwise there could be some vertex at *negative* distance from s.

The second vertex $v_2 = u$ must be the nearest out-neighbour of s: the vertex u with $(s, u) \in A$ and $\ell(s, u)$ minimal. Or, to be more precise, in case there are other vertices at the same distance, "we may take as v_2 the nearest out-neighbor u of s". Why? Any path out of s with cardinality at least 1 starts with some arc (s, v), and the total path length is at least $\ell(s, v)$, by the nonnegativity of arc lengths. So, if we are looking for a shortest path, it is enough to look at paths of cardinality 1 (a single arc out of v), and the shortest of these (the nearest out-neighbor u of s) is a shortest path.

The same reasoning applies in the general case, and here we'll be more formal. Assume that we already know the k nearest vertices, $R = \{v_1, v_2, \ldots, v_k\}$ — here R stands for "reached" — and the corresponding distances $d(v_i)$ (starting with $d(v_1) = 0$). How can we find the next vertex v_{k+1} and its distance $d(v_{k+1})$?

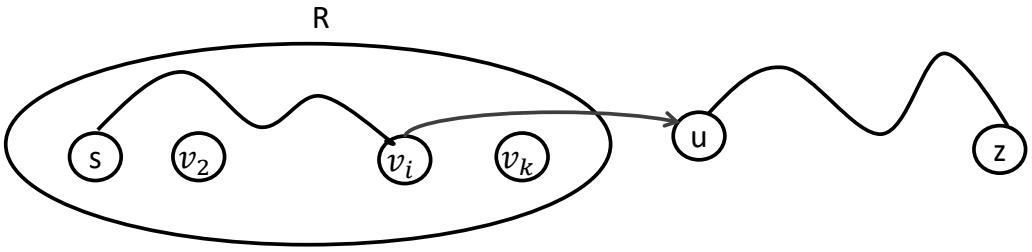


Figure 10.3: Illustration of finding a next-nearest vertex u.

With reference to Figure 10.3, let $z \in A \setminus R$ be a nearest vertex not yet reached, and let P be a shortest path from s to z. Since $z \notin R$, there must be some first vertex u in the path that is not in R, preceded by some vertex $v \in R$ (shown as v_i in the figure), so that $P = s, \ldots, v, u, \ldots, z$. By definition of P, $d(z) = d(v) + \ell(v, u) + \ell(u, \ldots, z)$. By the nonnegativity hypothesis, the last term is ≥ 0, so u is also a nearest vertex not yet reached. Also by definition of P, $s, \ldots, v, u$ is a shortest path to u (or else replacing it with a shorter path to u would give a shorter path to z). Thus, the nearest vertex (here, u) can be reached by a shortest path that follows a single arc out of a vertex (here, v) in R. So:

A next nearest vertex u can be found as one achieving the minimum in

$$\min\{d(v) + \ell(v, u) : v \in R, u \in A \setminus R, (v_i, u) \in A\}.$$

10.1.2 Description of Dijkstra's algorithm

Dijkstra's algorithm implements the above argument in a more efficient way. The main source of inefficiency above is that before a vertex u enters R, we always recompute $d(u)$ from scratch, by taking the minimum of $d(v_i) + \ell(v_i, u)$ for all arcs (v_i, u). For the sake of efficiency we will have temporary $d(u)$ values for vertices u outside R, updated gradually. Whenever a new vertex v_i is found, we compare $d(u)$ and $d(v_i) + \ell(v_i, u)$; if the latter value is smaller, we decrease $d(u)$ to this level. Hence we only need to consider every arc (v_i, u) once; $d(u)$ will reach its final value when u enters R.

As before, in Dijkstra's algorithm we maintain a set R of *reached vertices* where we already know the length of the shortest path, starting with $R = \{s\}$ and adding a new vertex in every step, terminating once t enters R (if we want only to find the shortest path length from s to t) or until all vertices are included in R (providing the shortest path lengths from s to every vertex).

We maintain the *distance estimate* $d(u)$ for every $u \in V$. For vertices $u \in R$, $d(u)$ will be equal to the length of the shortest s—u path. For $u \in V \setminus R$, $d(u)$ will be the shortest s—u path that is one hop from R: this is an upper bound on the true shortest s—u path, and is defined to be ∞ for a vertex that is not reachable by a single arc from R. We start by setting $d(s) = 0$, $d(u) = \ell(s, u)$ for every out-neighbour u of s, and $d(u) = \infty$ for all other vertices. Note that this is correct so far: d is the true distance for all vertices in R (i.e., for s), the shortest distance for vertices that are out-neighbors of R (that is, for vertices that are out-neighbors of s), and infinite otherwise.

Every iteration starts by expanding the set R by adding a vertex $u \in V \setminus R$ with minimum value $d(u)$. Then we update the estimates $d(v)$ on all out-neighbours v of u: for each such v, if $d(u) + \ell(u, v) < d(v)$, then we set the new value $d(v) = d(u) + \ell(u, v)$; otherwise, we leave $d(v)$ unchanged. This update maintains the property that $d(v)$ is the length of the shortest s—v path that is within R but for its final arc: the pre-update $d(u)$ is the shortest possible length where the final arc comes from a vertex in R other than u (assuming inductively that the algorithm was correct up to this point), while if the final arc does come from u the shortest possible path length is $d(u) + \ell(u, v)$.

Argue that this algorithm will add the vertices to R in the same order as the variant outlined above.

Identifying the shortest path

The algorithm computes the length of the shortest path from s to t; as argued above, the value $d(t)$ will return this value. We are not done yet: the true goal is to find a *shortest path*, not just its length.

One more ingredient is needed to achieve this. For every vertex v with $d(v) < \infty$ and $v \neq s$, we shall maintain a pointer $\text{pred}(v)$ pointing to the predecessor of v on the candidate shortest path. That is, whenever the value of $d(v)$ is updated to $d(u) + \ell(u, v)$, we also set $u = \text{pred}(v)$. The shortest s—u path can be found following these pointers from u back to s. A concise formal description is given in Algorithm 1.

Let us now verify that the s—t path returned by the algorithm is really a shortest one. Observe that whenever $u = \text{pred}(v)$, we must have $d(u) + \ell(u, v) = d(v)$. Let $u_0 u_1 \dots u_k$

Algorithm 1 Dijkstra's algorithm

INPUT: Directed graph $G = (V, A)$, vertices $s, t \in V$, nonnegative length function $\ell : A \to \mathbb{R}_+$.

OUTPUT: A shortest path from s to t.

1. Set $d(s) = 0$ and $d(u) = \ell(s, u)$ whenever $(s, u) \in A$; set $d(v) = \infty$ for all other vertices.
2. Set $\text{pred}(u) = s$ whenever $(s, u) \in A$.
3. Set $R = \{s\}$.
4. While $t \notin R$ repeat
   (a) Pick $u \in V \setminus R$ as a vertex having minimum $d(u)$ value; break ties arbitrarily.
   (b) Include u in R.
   (c) For every $v \in V \setminus R$ with $(u, v) \in A$ and $d(u) + \ell(u, v) < d(v)$ do
      i. Set $d(v) = d(u) + \ell(u, v)$.
      ii. Set $\text{pred}(v) = u$.
5. Follow the pointers back from t to $\text{pred}(t)$, $\text{pred}(\text{pred}(t))$ and so on, until s is reached.
6. **Return** the corresponding path.

denote the path P returned by the algorithm with $u_0 = s$ and $u_k = t$. We obtained every vertex u_{i-1} as the predecessor of u_i for $i = 1, \ldots, k$, and hence $d(u_i) = d(u_{i-1}) + \ell(u_{i-1}, u_i)$ must hold. Let us write down all these equalities:

$$\begin{aligned} d(u_1) &= d(u_0) + \ell(u_0, u_1) \\ d(u_2) &= d(u_1) + \ell(u_1, u_2) \\ &\vdots \\ d(u_k) &= d(u_{k-1}) + \ell(u_{k-1}, u_k). \end{aligned}$$

Adding up all these equalities, $d(u_1), d(u_2), \ldots, d(u_{k-1})$ appear on both sides exactly once. Since $u_0 = s$, we have $d(u_0) = 0$, and since $u_k = t$, we can replace $d(u_k)$ by $d(t)$. Hence we obtain

$$d(t) = \sum_{i=0}^{k-1} \ell(u_i, u_{i+1}) = \ell(P).$$

This means that P does have length exactly $d(t)$, i.e., P is a shortest path from s to t. This completes the proof of the correctness of Dijkstra's algorithm.

10.1.3 Example

We now demonstrate how the algorithm works on the example of Figure 10.2. The iterations of the algorithm are shown in Figure 10.4. The set R is represented by the red vertices,

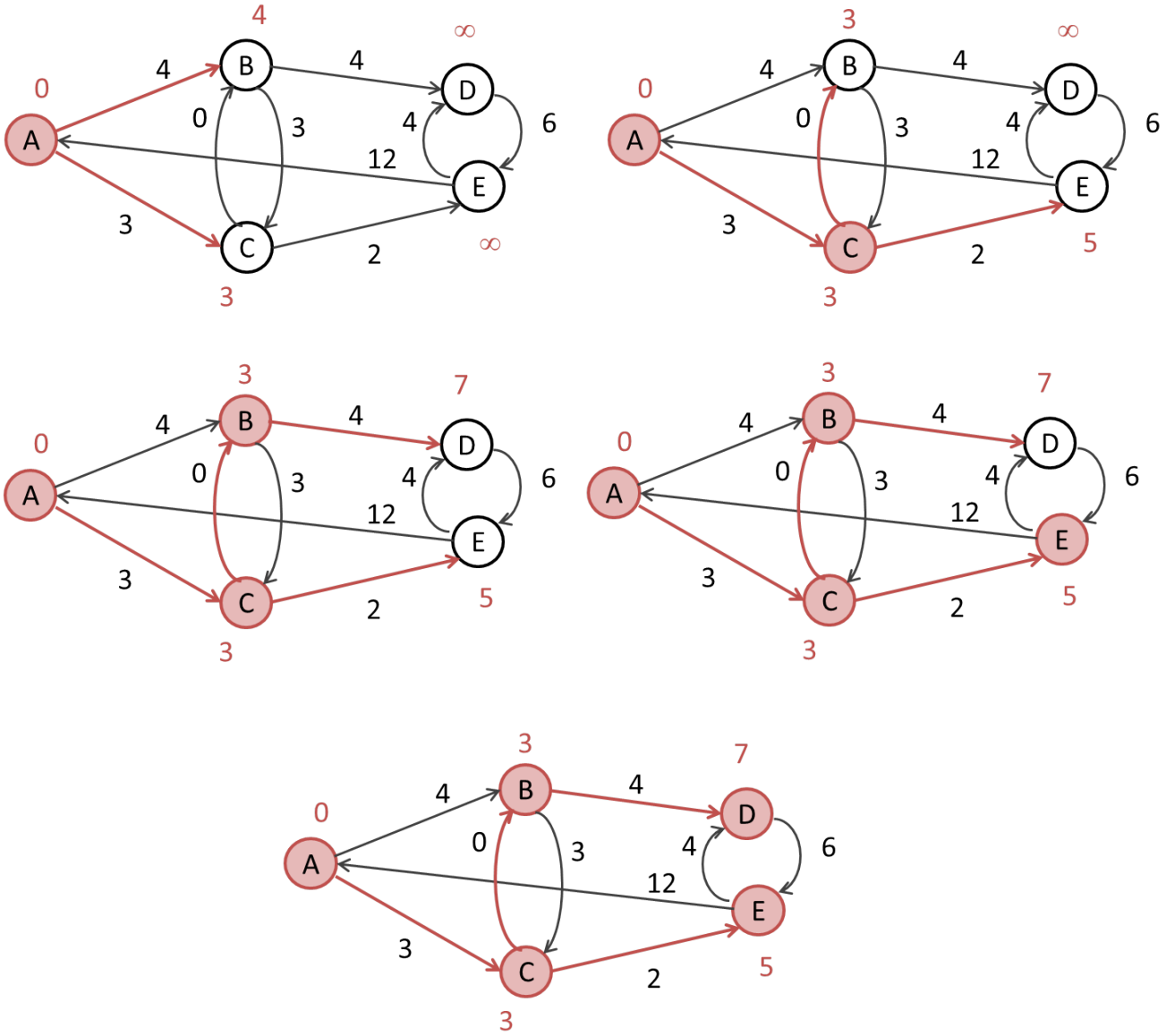


Figure 10.4: Illustration of Dijkstra's algorithm.

and the red arcs represent the predecessor pointers: the arc (u, v) is red if $\text{pred}(v) = u$. At initialisation, R consists of the single vertex A, the starting point (denoted by s in the algorithm). The algorithm terminates in iteration 5 when the target vertex D (denoted by t in the algorithm) enters the set R. In this example all other vertices enter before t, but it is only by chance that t is last; the algorithm may terminate with many vertices still outside R.

The following table gives the $d(u)$ values and predecessors in every iteration; bold numbers represent vertices in R. A detailed explanation of every step is given below.

Iterations	1		2 : enters C		3 : enters B		4 : enters E		5 : enters D	
	$d(u)$	$\text{pred}(u)$	$d(u)$	$\text{pred}(u)$	$d(u)$	$\text{pred}(u)$	$d(u)$	$\text{pred}(u)$	$d(u)$	$\text{pred}(u)$
A	**0**	–	**0**	–	**0**	–	**0**	–	**0**	–
B	4	A	3	C	**3**	C	**3**	C	**3**	C
C	3	A	**3**	A	**3**	A	**3**	A	**3**	A
D	∞	–	∞	–	7	B	7	B	**7**	B
E	∞	–	5	C	5	C	**5**	C	**5**	C

Iteration 1 At initialisation, we set $d(A) = 0$, and for the two out-neighbours B and C of A, we set labels $d(B) = \ell(A, B) = 4$ and $d(C) = \ell(A, C) = 3$. The remaining two labels are $d(D) = d(E) = \infty$. We also set the predecessors $\text{pred}(B) = A$, $\text{pred}(C) = A$.

Iteration 2 We choose the vertex with the smallest label outside R. This will be C, hence we get $R = \{A, C\}$. The labels of the two out-neighbours of C have to be updated. We set $d(E) = d(C) + \ell(E, C) = 3 + 2 = 5$, and $\text{pred}(E) = C$. What happens to B is more interesting. Currently $3+0 = d(C)+\ell(C, B) < d(B) = 4$, hence we decrease $d(B)$ from 4 to 3, and also modify the predecessor to $\text{pred}(B) = C$. In fact, the path A–C–B is of length 3, shorter then the single-arc path A–C of length 4.

Iteration 3 B is added to R, as it has the smallest d value: we get $R = \{A, B, C\}$. B has two out-neighbours, C and D, but C is already contained in R. Hence we need to update only the label of D: this gives $d(D) = d(B)+\ell(B, D) = 3+4 = 7$, and set $\text{pred}(D) = B$. Note that the red arcs already include the path A–C–B–D that will prove to be the shortest A—D path, but the algorithm is not yet finished, and in principle we might still find a shorter path.

Iteration 4 E is added to R, giving $R = \{A, B, C, E\}$. We need to examine the out-neighbour D of E. Since $7 = d(D) \leq d(E) + \ell(E, D) = 5 + 4 = 9$, we do not modify any labels or predecessors.

Iteration 5 In the final iteration, we add the remaining vertex D. No labels or predecessors need to be modified. The algorithm terminates as we have reached the target vertex.

Finally, we can find the shortest path from A to D by following the red arcs denoting the predecessors. From D we follow the arc to its predecessor B, then to C, and finally A. In the forward direction, this gives the path A–C–B–D of length 7.

10.2 Remarks

Physical intuition

Dijkstra's algorithm can be visualised in the following manner. Consider the special case when the costs are symmetric: the arcs come in pairs (u, v), (v, u), and $\ell(u, v) = \ell(v, u)$. Model the vertices of the graph by balls, connected by strings, with the string between the balls corresponding to u and v having length $\ell(u, v)$. Put the balls on a table, and slowly start raising the ball corresponding to the starting vertex s. The balls will leave the table in the order they enter the set R. When the procedure ends, the height difference between a ball u and the highest ball s will be exactly the length of the shortest path. Moreover, the shortest path will consist of tight strings; there can be some loose strings that do not participate in the shortest paths.

Single source shortest paths

The algorithm computes not only a shortest path from s to t, but from s to every vertex in the set R. The distinguished role of t in the algorithm is only in the termination criterion: we stop once t enters R. We could instead keep the algorithm running until the entire V is exhausted, in which case we achieve something remarkable: the shortest paths from s to every vertex of the graph. (The predecessor arcs give a "shortest path tree".) If we are only interested in a single vertex t, we might hope that the algorithm terminates earlier, but there is no guarantee that t won't be last, and "on average" (suitably defined) t occurs half way through, as you would expect.

Questions of implementation

The description above gives all details of Dijkstra's algorithm. However, if you had to implement it on a computer, there are several technical issues to be decided. To start with, how is the graph represented? How do we identify the vertex in $V \setminus R$ that minimises $d(u)$? Various different data structures and subroutines can be applied; the running time of the algorithm may hugely depend on these choices. These questions are beyond the scope of the current course; you may find out more about these in, e.g., Chapter 4 of [AMO].

Nonnegativity

Let us now reflect on an important hypothesis.

Question: Why do we require that all arc costs are nonnegative?

We used this hypothesis repeatedly in the proof of correctness, for example when assuming that the trivial length-0 path from s to itself is a shortest path, and in reasoning that the length of the part of path P between z and u was nonnegative. Without the nonnegativity assumption, the algorithm does not work correctly. Figure 10.5 gives an example of such behaviour; let A be the starting point and B the target point. At initialisation, the algorithm sets $R = \{A\}$, $d(B) = 2$, and $d(C) = 3$. The next iteration will add B to R, since $d(B) <$

$d(C)$. B has no out-neighbours, hence no distances d change. The algorithm terminates here as B is reached. However, the shortest path from A to B is A–C–B, with length $3 + (-2) = 1 < d(B) = 2$.

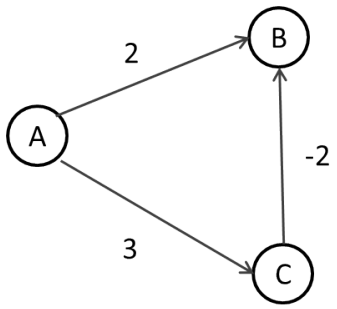


Figure 10.5: An example with negative weights where Dijkstra's algorithm fails.

Given the failure of Dijkstra's algorithm, is it possible to find shortest paths efficiently, if negative lengths are allowed? Surprisingly, this makes the problem substantially harder. There is no "efficient algorithm" known for the problem, one that always returns the correct answer in a reasonable amount of time, for all input graphs and arc lengths. The existence of such an algorithm is considered highly unlikely and would have "magical" implications in computer science; see Chapters 7-9 in [Pa] for the background in computational complexity. However, in the next lecture we shall see a special class of graphs where the shortest path problem with negative lengths can be solved.

Suggested reading

Section 9.3 of [HL] (both 7th and 9th eds) gives a brief description of the shortest path problem. More comprehensive treatments are given in Section 24 of [C] and in Section 4 of [AMO].

10.3 Exercises

Exercise 10.1. Use Dijkstra's algorithm to find a shortest path from s to t in the grid network in Figure 10.6.

Exercise 10.2. The following table illustrates the possible shifts for the drivers of a bus company. We wish to ensure, at the lowest possible cost, that at least one driver is on duty for each hour of the period 9am to 5pm. Formulate and solve this as a shortest path problem.

Duty hours	9–1	9–11	12–3	11–3	2–5	1–4	4–5
Cost	30	18	21	38	20	22	9

(Note that your model should allow shifts to overlap. For example, the combination of shifts 9-1, 12-3, and 2-5 should be allowed, as the three shifts cover the 9-5 period.)

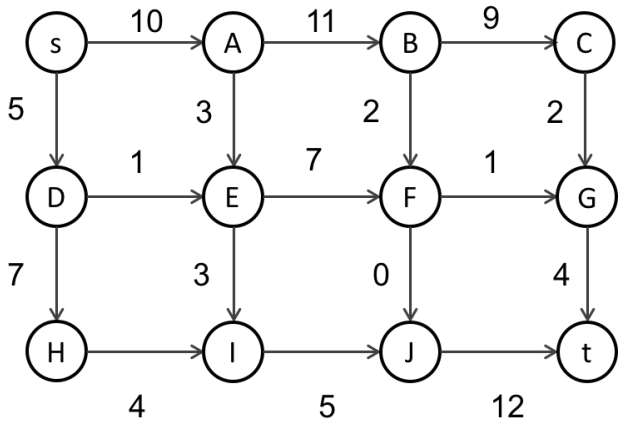


Figure 10.6: Network of Exercise 10.1

Exercise 10.3. *(Exam question 2015.)* A hospital has a PET scanner that requires frequent maintenance. The management wishes to identify the best maintenance plan for the next 4 years. The unit is new at the beginning of year 1. At the beginning of every subsequent year, they have the option of sending back the unit to the manufacturer for a major overhaul, restoring it to the original condition. The overhaul cost is expected to increase in the future. It would cost £10,000 at the beginning of year 2, £11,000 at the beginning of year 3, and £12,000 at the beginning of year 4.

Maintenance has to be performed every year. The table below shows the expected maintenance costs depending on the time since purchase or the last overhaul. For example, if an overhaul is performed at the beginning of year two (for £10,000), then maintenance in year two will cost £3,000. Advise the hospital management on the schedule of overhauls that minimises the expected total costs over the 4 years. To solve the problem, formulate it as a shortest path problem and apply Dijkstra's algorithm. *(Hint: nodes can correspond to different states, representing the actual year and the number of years since the last overhaul.)*

Years since purchase or overhaul	Expected cost of maintenance
0	£3,000
1	£5,000
2	£8,000
3	£12,000

Exercise 10.4. *Prove or refute the following claim:* Let P be a shortest path from s to t, and let u be an arbitrary vertex between them on the path P. Then the part of P from s to u is a shortest path from s to u.

Exercise 10.5. A certain telecommunication network is modelled by a directed graph. Data transmission is very quick, so finding shortest paths is not important. However, the connections are prone to failure: there is a probability p_{ij} that link (i, j) fails while transmitting. These failures occur independently from one other. Data going from a source s to a target

t is routed along a path, and the data arrives if none of the links along the path fail. The objective is to find the most reliable s—t path, the one minimising the failure probability. Solve the problem by reducing it to the shortest path problem, by defining an appropriate length function.

Chapter 11

Project management via Critical Path Analysis

A major project such as a construction project consists of several different activities. Network optimisation is a standard tool to optimally schedule these activities and gain further insights into the project design. *CPA* (Critical Path Analysis) and *PERT* (Programme Evaluation and Review Technique) are two very similar OR methodologies developed in the late fifties, that have since been merged into a technique called CPA/PERT that continues to be widely used today. Here we describe the core parts of the method, and refer to it as CPA for simplicity. As we shall see, it is closely related to the shortest path problem.

Let us start with an example. A construction company is contracted to build a house within 48 weeks. The construction process is decomposed into seven activities such as putting up the rough wall or plumbing. The duration of every activity is estimated in weeks. Some activities can only be performed after others are finished, whereas other activities can be carried out simultaneously. The table below shows a (simplified) list of activities, with the last column showing the immediate prerequisites.

The **immediate prerequisites** of an activity are the other activities that must be finished no later than the activity is started, with no third activity that must be necessarily carried out between the two.

For example, putting up the rough walls (B) is an immediate prerequisite of plumbing (C). Note that laying the foundation (A) is also a prerequisite of plumbing (B), but not an immediate one: activity B must always be carried out after A and before C.

Activity	Description	Estimated Duration	Immediate Prerequisites
A	Excavate & lay the foundation	6 weeks	–
B	Put up the rough wall	10 weeks	A
C	Plumbing	8 weeks	B
D	Do the electrical works	6 weeks	C
E	Put up the roof	6 weeks	B
F	Put up the siding	9 weeks	E, C
G	Interior works	12 weeks	D, F

The main objective of the company is to find an optimal timing of these activities to meet the deadline of 48 weeks. A convenient way to represent the process is using a **project network**. In this directed graph, the vertices correspond to the activities, and if activity X is an immediate prerequisite of activity Y, then we add an arc (X, Y). Further, we add two artificial new vertices to the graph, s representing the start and t the termination of the works. We connect s with an arc to every activity that has no immediate prerequisite, and add an arc to t from every activity that is not an immediate prerequisite of any further activity.

Length values on the arcs correspond to activity durations, so where we previously used ℓ for arc "length" we will now use $\tau(u, v)$ (τ being the Greek letter t, evoking "time"). For every arc (X, Y), we set $\tau(X, Y)$ equal to the duration of X. The length on every arc leaving the starting point s will be 0. Our aim is to understand how long the project might take. The project network for our construction work is given in Figure 11.1.

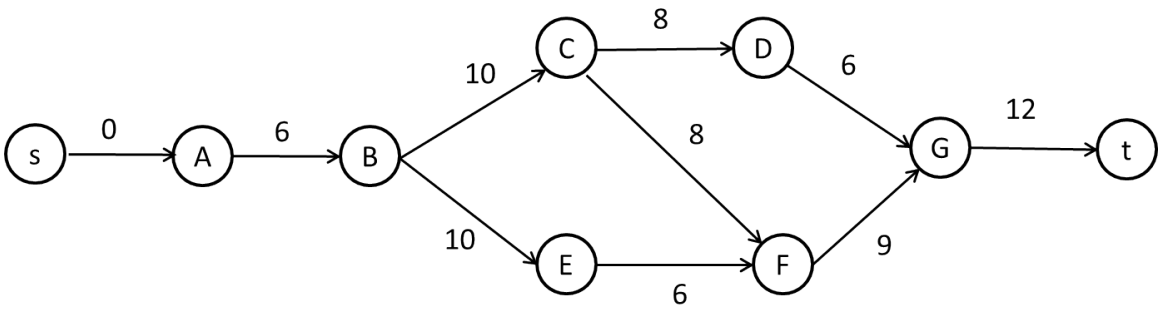


Figure 11.1: Construction project network.

11.1 Longest paths

Consider an arbitrary path from s to t, for example s–A–B–C–D–G–t. This corresponds to a sequence of activities that cannot be parallelised: they must be carried out strictly in this order. The length of this path is 42. This shows that the whole project will require at least 42 weeks in total. The same argument works for all other paths as well (there are three in all): the path s–A–B–E–F–G–t gives 43 weeks, whereas s–A–B–C–F–G–t gives 45 weeks.

We may therefore conclude that the project cannot take less then 45 weeks. The following theorem asserts that the best bound we can obtain from a path gives the optimal length of the project.

Theorem 11.1. *The minimum time required to complete the project equals the maximum length of an s—t path in the project network.*

The CPA algorithm performs two tasks simultaneously, thereby verifying this theorem:

(i) finding an optimal schedule of activities, and

(ii) finding a longest path in the project network of the same length as the optimal schedule.

11.1.1 Connection to shortest paths

Before presenting the CPA algorithm, we explain how it is related to the shortest path problem from the previous lecture. The length of an arc $\tau(u, v)$ in the project network equals the time needed to complete activity u. Let us define a second length function, that is the negative of τ: let $\ell(u, v) = -\tau(u, v)$. The following claim is straightforward.

Proposition 11.2. *The length of a* longest *s—t path for the length function τ equals the negative of the length of the* shortest *path for the length function $\ell(u, v)$, where $\ell(u, v) = -\tau(u, v)$ for every arc.*

However, Dijkstra's algorithm is *not* applicable for this problem: recall that the arc lengths were assumed to be nonnegative, and we saw an example with negative arc lengths when the algorithm fails. In fact, there is strong theoretical evidence indicating no efficient algorithm can be expected for solving the shortest path problem in arbitrary graphs and arbitrary arc lengths (specifically, negative lengths). Indeed, one way to view the difficulty is that negating all lengths turns a shortest-path problem into a longest-path problem, and where it is "easy" to find shortest paths (Dijkstra's algorithm is efficient), it appears to be "hard" to find longest paths. (There is reason, based on the theory of computational complexity, to believe that there is no efficient algorithm for this or the related "Hamilton path" problem, but this is outside the scope of the present lectures.)

Instead of restricting lengths to be nonnegative, an alternative is to allow all lengths but impose a restriction on the network structure. This is exactly what will happen in the CPA algorithm: project networks admit a certain *"natural" ordering*, that gives rise to an algorithm even simpler than Dijkstra's. We present the algorithm directly for longest paths, but it can easily be modified to a shortest path algorithm for graphs with natural orderings (and arbitrary weights).

11.1.2 Natural orders

A special property of project networks is that the vertices admit the following:

Let $G = (V, A)$ be a directed graph, with the vertices ordered as $v_1, v_2, \ldots, v_n$. This is called a **natural order** of the vertices if for any arc $(v_i, v_j) \in A$, it holds that $i < j$.

For a project network, it means that we can write activities in a certain order so that all immediate prerequisites of an activity precede it in the ordering. In particular, s must be the first and t the last vertex in such an order. As an example, consider the project network of Figure 11.1. The way it is drawn immediately gives a natural order from left to right; we only need to break the ties (between C and E and between D and F). One natural order is $s, A, B, C, E, D, F, G, t$; another one is $s, A, B, E, C, F, D, G, t$. On the other hand, $s, A, B, E, F, C, D, G, t$ is *not* a natural order, since C is an immediate prerequisite of F yet F precedes C in the order.

Natural orders are also known as *topological orders*. The reason behind the existence of such an order is that the project network cannot have any directed cycles. (A network without directed cycles is called a *directed acyclic graph*, or *DAG*.) That is, we cannot have a sequence of activities $u_0, u_1, \ldots, u_{k-1}$ with arcs (u_0, u_1), (u_1, u_2), …, (u_{k-1}, u_k), and (u_k, u_0). Such a cycle would mean that u_0 must be completed before u_1 is started, u_1 must be completed before u_2 is started, etc., and u_{k-1} must be completed before u_0 is started. Consequently, u_0 would have to be completed before u_0 is started: a contradiction. Therefore any directed cycle in the project network indicates an error in the project description, making it impossible to schedule. In Exercise 11.5 you are asked to show the converse.

11.2 The CPA algorithm

In the CPA algorithm, we will always assume that a natural ordering is given on the vertices of the project network. (We can safely assume this because there are very efficient algorithms for topologically sorting any DAG; for small networks it be done by inspection.) Let this order be $v_1, v_2, \ldots, v_n$ with $v_1 = s$ and $v_n = t$. The CPA algorithm is actually simpler than Dijkstra's algorithm; in the latter we needed to compute certain distance estimates to determine the next vertex, whereas in CPA we just add the vertices in the natural order. We let $p(v_i)$ denote the *earliest possible start time* of the ith activity in the order. Clearly, $p(s) = 0$. For the i^{th} vertex, we compute

$$p(v_i) = \max\{p(v_j) + \tau(v_j) : 1 \leq j < i, (v_j, v_i) \in A\}.$$

That is, we take all immediate predecessors v_j of v_i. We have already computed the earliest start time $p(v_j)$, and to this we add $\tau(v_j, v_i)$, the time needed for activity v_j to be completed. This gives a lower bound on when v_i can start, and we take the largest among these lower bounds.

The longest path in this context is called the *critical path*: recall that CPA stands for Critical Path Analysis. We use pointers $\text{pred}(u)$ the same way as in Dijkstra's algorithm to identify longest paths. Algorithm 2 gives the formal description. We allow arbitrary length

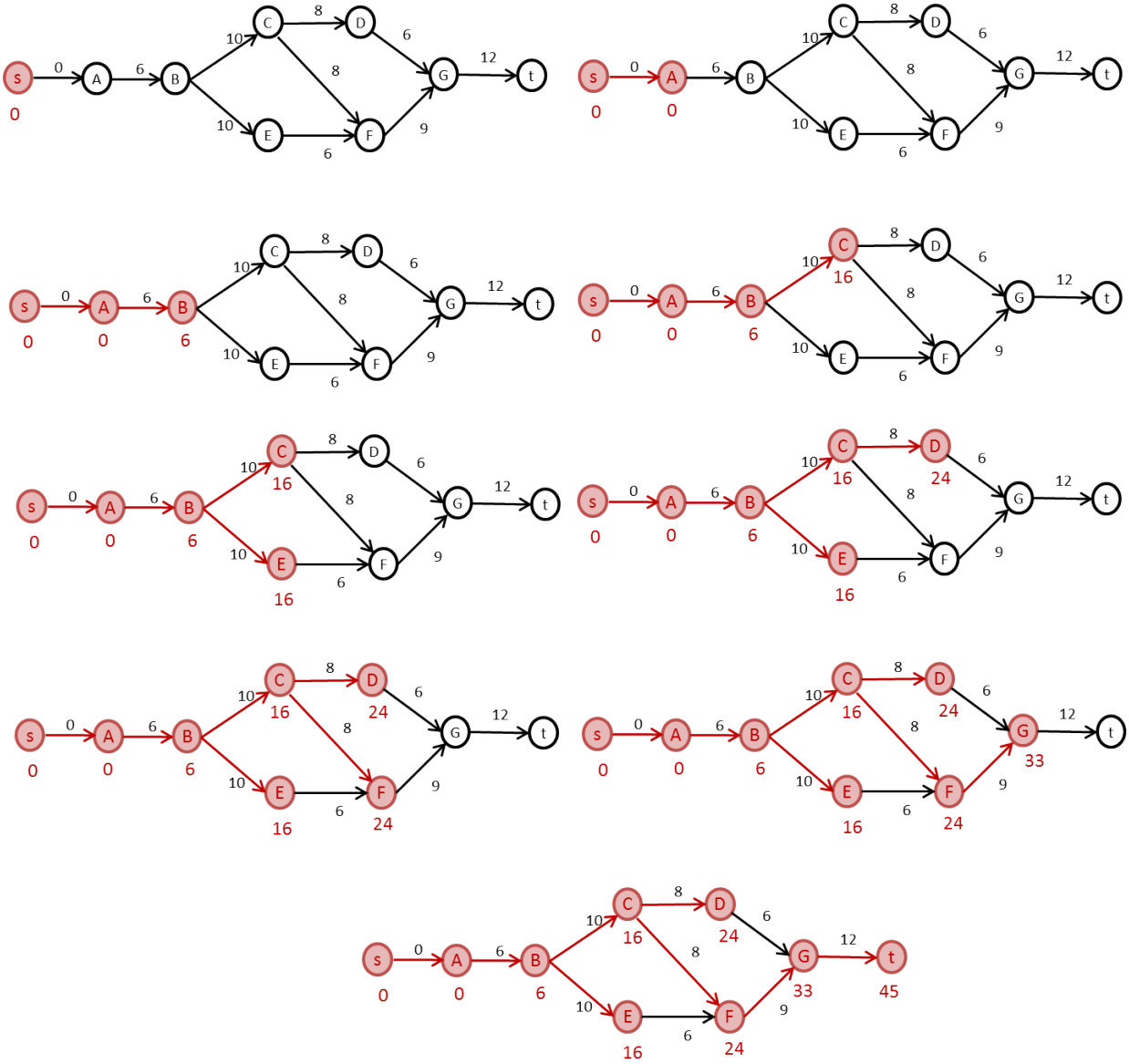


Figure 11.2: Application of the CPA algorithm to the problem instance of Figure 11.1.

Algorithm 2 CPA

INPUT: Project network $G = (V, A)$ with a natural ordering $v_1, v_2, \ldots, v_n$, with $v_1 = s$, $v_n = t$. and an arbitrary length function $\tau : A \to \mathbb{R}_+$.

OUTPUT: A critical path from s to t and the starting times of the activities in an optimal schedule.

1. Set $p(s) = 0$.
2. For $i = 2, 3, \ldots, n$ do
   (a) Set $p(v_i) = \max\{p(v_j) + \tau(v_j) \colon 1 \leq j < i, (v_j, v_i) \in A\}$.
   (b) Set $\text{pred}(v_i) = v_j$ to the value j giving the maximum value above; break ties arbitrarily.
3. Trace back from t to $\text{pred}(t)$, $\text{pred}(\text{pred}(t))$, and so on, until s is reached. This gives the critical path.
4. **Return** the critical path that corresponds to the arcs found in tracing back, and, for every activity v_i, the starting time $p(v_i)$.

functions, including negative lengths; indeed, although τ is defined as nonnegative for project networks, the algorithm works more generally.

In Figure 11.2 we demonstrate the algorithm on the project network of Figure 11.1, using the natural ordering $s, A, B, C, E, D, F, G, t$. The red numbers represent the p values, and the red arrows the precedence relation. Most iterations are trivial, as all vertices other than F and G have only a single incoming arc. In iteration 7, C was chosen as the predecessor rather than E since $16 + 8 > 16 + 6$. Similarly, in iteration 8, we had to compare $24 + 6$ and $24 + 9$.

Hence we identified the critical path s–A–B–C–F–G–t. The starting times for the activities are simply given by the $p()$ values. To translate it to the actual schedule, we use the following convention. We assume that all activities start on Monday morning, the first denoted as Week 1; thus time $p() = 0$ means Monday morning of week 1, and time $p(v) = h$ means Monday of week $h + 1$. For example, activity A starts at time $p(A) = 0$ and runs for 6 weeks, and therefore we schedule it for Weeks 1–6; activity G starts at $p(G) = 33$ and requires 12 weeks, therefore it runs for Weeks 34–45. The algorithm guarantees that every activity scheduled this way will start only after all its immediate prerequisites have been completed. Recall that the project deadline required completion in 48 weeks, and note that this solution meets the deadline.

Activity	Description	Scheduled weeks
A	Excavate & lay the foundation	1–6
B	Put up the rough wall	7–16
C	Plumbing	17–24
D	Do the electrical works	25–30
E	Put up the roof	17–22
F	Put up the siding	25–33
G	Interior works	34–45

11.3 Floats of activities

In construction work, the time estimates might be very rough, as there is a lot of uncertainty in weather conditions, labour availability, arrival of ordered materials, etc. Also, in the above example, every estimate was given in weeks: some activities may require a few days more or less. One may therefore ask how realistic it is to use the schedule obtained by the above method, and how changes affect the completion time of the project. This is similar, and indeed related, to sensitivity analysis in Linear Programming.

Consider activities on the longest (critical) path s–A–B–C–F–G–t. If any of these activities takes longer than expected, it affects the entire outcome, hence the term *critical* path. For example, if plumbing (activity C) takes a week longer, and putting up the siding (F) three weeks longer, then performing activities A–B–C–F–G will require 49 weeks instead of 45, thereby failing to meet the deadline.

On the other hand, there is some flexibility for other activities. For example, putting up the roof (E) needs to be completed only by the time putting up the siding (F) starts. Currently E is scheduled to finish by the end of week 22, leaving a "slack" or *free float* of 2 weeks as it only needs to be completed by the end of week 24 for its successor activities (here only F) to start on time. This float provides various options: we can preserve the slack so that time over-runs in E will not delay the whole project; we can postpone E's start time; or we can possibly cut costs for E by allowing it to take longer (for example by employing less-skilled workers). Similarly, activity D (the electrical works) has a slack of 3 weeks.

It is important to understand how much scope is there for flexibility with the individual activities. Let us recall that:

> For an activity u, the starting time $p(u)$ obtained by the CPA algorithm represents the *earliest possible start time* for the activity u.

In what follows, we describe a method to obtain the *latest possible start time* $q(u)$ for each activity u by running the algorithm once more.

Obtaining latest possible start times.

1. After the initial run of the CPA algorithm, record the longest path length $L = p(t)$.
2. Reverse all arcs of the graph, and correspondingly reverse the natural order (to start with t and end with s).
3. Run the CPA algorithm on the modified instance starting from the first vertex t; let $r(u)$ denote the labels obtained by the algorithm.
4. Set $q(u) = L - r(u)$ as the latest possible start time.

In the above example, we had $L = 45$. The table below shows both the $r(u)$ values and the $q(u)$ values. The interpretation of $r(u)$ is that it is the least amount of time that has to pass between the start of u and the termination of the project; thus $q(u) = L - r(u)$ is the latest possible starting time for u. The table's last column denotes the *(total) float*, the difference $q(u) - p(u)$. This is the longest time by which the activity can be delayed, or prolonged, while still completing the project within the originally scheduled time L.

Activity	Description	$p(u)$	$r(u)$	$q(u)$	float
A	Excavate & lay the foundation	0	45	0	0
B	Put up the rough wall	6	39	6	0
C	Plumbing	16	29	16	0
D	Do the electrical works	24	18	27	3
E	Put up the roof	16	27	18	2
F	Put up the siding	24	21	24	0
G	Interior works	33	12	33	0

Note the difference between the "free float" seen earlier and the "total float" here. Free float is the extra time an activity may take without delaying any of its successor activities (so the earliest possible start time of each is unaffected). Total float is the extra time an activity may take without delaying the project's completion time. To see the difference, imagine some activities X and Y preceding Z, with Y having no other successor, and neither Y nor Z on the critical path. Suppose Z has a free float of 2, and that its earliest possible start time $p(Z)$ is dictated by X, with Y finishing 1 time unit earlier. Then the free float of Y is 1, because it can finish that much later without delaying any of its successors (i.e., just Z), but its total float is at least $1 + 2 = 3$, because a delay of 3 in finishing Y means a delay of 2 in finishing Z, which is within Z's free float, thus giving no delay to any further activities, thus no delay to completing the whole project.

Variant. To compute the $r(u)$ values, you do not necessarily need to construct the reverse network, but can adapt the algorithm directly to the original network, starting at t rather than s, following arcs backward rather than forward, and so on. *How exactly?*

11.4 Remarks

In the project-network construction we used what is called the *activity-on-arc (AOA)* convention: each activity's duration is assigned to the length of every arc leaving the activity's vertex. An equivalent representation is the *activity-on-node (AON)* convention: each activity's duration is assigned to the activity's vertex (the terms "vertex" and "node" are used interchangeably), and no length is associated with the arcs, which merely represent the immediate prerequisites. The CPA algorithm can be naturally adapted to AON networks. *(How?)* We chose the more traditional AOA representation to emphasise the connection to shortest path computations, but AON networks are more commonly used nowadays since they are easier to represent and modify, and easier for managers to understand.

Another way to deal with project time uncertainties is to treat them as random variables. The PERT method also includes ways to estimate the *distribution* of the total project length.

A further important ingredient of the CPA method is *crashing activities*. Assume the schedule obtained takes longer than the feasible time horizon for the project, but that some activities can be rushed, for an additional cost. The goal is to expedite activities at a minimum cost to achieve the desired project length. See the suggested readings for more detail.

Suggested reading

Chapter 10 in ed. 7 of [HL] gives a detailed description of the method, including the time uncertainties in Sec 10.4 and crashing activities in Sec 10.5. The end of the chapter also includes case studies. In ed. 9, a less detailed description of the entire method is in Sec 9.8.

11.5 Exercises

Exercise 11.1. Setting up a new TV channel for TV station CCB involves a number of activities. These are identified below, together with the cost per day for each activity (in £'000) and its normal time to completion (in days). How long will it take to set up the channel? What are the critical paths, and the total floats on the non-critical activities?

Activity	Prerequisite	Days Required	Cost per Day
A	–	2	150
B	–	5	80
C	A, B	5	100
D	B	4	60
E	C	6	70
F	D	2	130
G	–	8	110
H	G, D	6	90
I	E, F	4	80
J	E, F	2	70
K	H, J	5	120
L	I	4	90

The work is to be carried out by London Telecoms Limited ("LTL"), and on successful completion CCB will pay them £6 million if they finish the project in 25 days or longer, and an extra £150,000 for every day by which the project makespan is less than 25 days. How much profit can LTL make?

LTL could purchase two extra person-days of consultancy help for £120,000 per day in order to reduce the makespan. Each extra person-day spent on an activity will reduce its duration by one day. For example, they could pay to save one day on each of activities A and D, or two days on activity F, although these would not be optimal choices. Why not? Would it be sensible to save two days on activity I? How best could LTL invest these extra two days?

Exercise 11.2. Find the optimal (quickest) schedule and floats for the activities given in the table. Besides the time constraints, a further restriction could be that the total number of workers available is limited at every point of time. Assume that the project below has to be carried out with a workforce limit of 6 people per day. Are you able to reschedule the activities that can be floated to keep the schedule optimal, but also respect the workforce limit? If not, how much more time will be needed?

Activity	Dependent on	Days	People Required per Day
A	–	2	4
B	A	4	3
C	A	6	3
D	B, C	8	1
E	B	3	5
F	C	3	4
G	D, E, F	2	5

Exercise 11.3. The CPA algorithm finds a critical path, but there might be multiple critical paths in a network. How you can decide if there is more than one critical path and, if so, exhibit a second one? For a given activity, can you decide if there is any critical path containing this particular activity?

Exercise 11.4. *(Exam question 2016.)* Edgar owns a house on the Isle of Wight and wishes to rent it out for the summer. Every rental period starts at 3pm on the day of arrival, and ends at checkout at noon of the day of departure. Using a specialised website, Edgar solicits a number of bids. Each bid specifies a rental period (by its arrival and departure date), and the total amount of money offered for this period. Edgar wishes to identify a set of bids that maximise his total revenue. Of course, he cannot accept bids for overlapping periods, but an accepted bid can start on the same day (from 3pm) as the previous one ends (at noon).

Explain how this problem can be formulated as a longest path problem. What algorithm would you suggest to find an optimal solution efficiently?

Exercise 11.5. Verify that the vertices of a directed graph have a natural ordering if and only if the graph contains no directed cycle.

1. First prove the easier direction: show that if there is a natural order then there is no directed cycle.

2. Then prove that if there is no directed cycle then there is a natural order. To do so, prove the following claim: for an arbitrary subset $S \subseteq V$ of the vertices, there must exist a vertex $u \in S$ that has no in-neighbours inside S. Use this claim to prove the main statement (hint: think of u as a "first" vertex in S).

Chapter 12

Dynamic Programming

12.1 Introduction

Dynamic Programming (DP) is a general mathematical technique to solve complex optimisation problems by determining the optimal sequence of interrelated decisions. The problem is decomposed into smaller subproblems, and we build up the solution by devising the optimal decisions stage-by-stage. In contrast to linear programming (LP), there is no standard form of dynamic programming. DP is a flexible approach, that can be applied in different contexts. We introduce DP and its main characteristics via a series of examples.

12.1.1 The Stagecoach Problem

The Stagecoach Problem is a standard illustration of DP. It is a shortest path problem, where a 19th century traveller has to reach from his embarkation point A (Missouri) to his destination F (California) across the dangerous wild west (see Figure 12.1). The possible intermediate stops decompose into layers, called *stages*. The numbers on the edges represent the lengths of these routes. The traveller wishes to find a shortest route.

In Chapter 10 we learned Dijkstra's algorithm to solve shortest path problems. We now solve the Stagecoach Problem via DP. You will see that in this example, the DP algorithm is similar to Dijkstra's algorithm, although it proceeds backwards, from the sink towards the source. DP is a more general principle applicable to a wide range of problem types. The purpose of the Stagecoach example is to illustrate the main characteristics of DP in a setting you are already familiar with.[1]

The DP algorithm for the stagecoach network identifies the shortest path from each node to the sink F, proceeding backwards through the layers. We start with the three *states* (the DP terminology for nodes) in stage E. What is the cost of the shortest routes from each of $E1$, $E2$, and $E3$ to F? This is easy: 7, 5, and 5 respectively.

Let us now move on to stage D. What are the shortest routes from $D1$, $D2$, and $D3$ to F? From $D1$ we have 3 choices for the next step:

[1] We note that the shortest path network in the Stagecoach Problem has a specific structure: it decomposes into layers, and one can only move from a node in one layer to a node in the next layer. Recall that no such assumption was made for Dijkstra's algorithm: it can compute shortest paths in directed graphs of arbitrary structure.

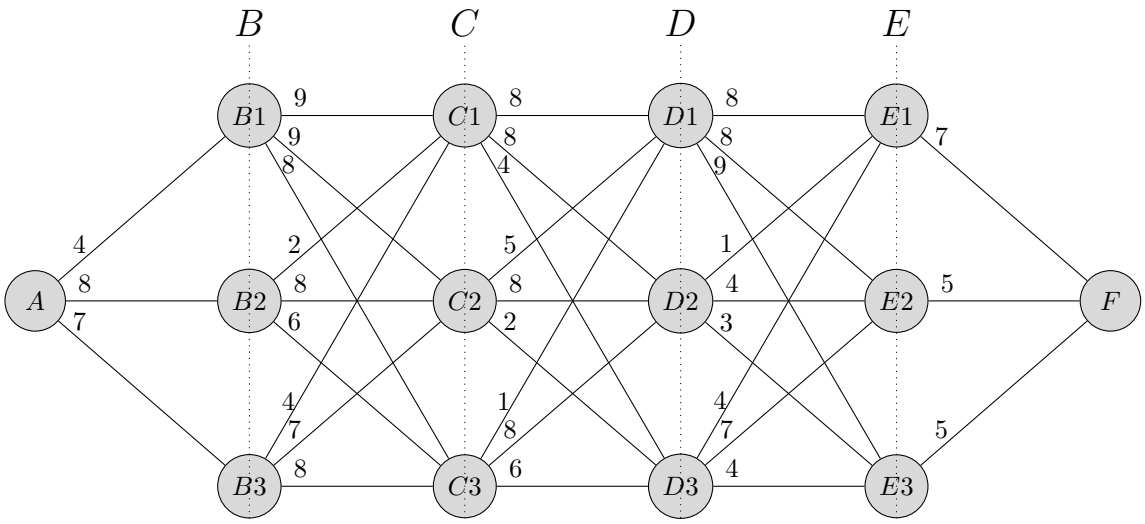


Figure 12.1: Network of the Stagecoach Problem

- we can go to $E1$. This involves a distance of 8, and the minimum distance from $E1$ to F is 7. Hence the minimum distance from $D1$ via $E1$ to F is $8 + 7 = 15$.
- we can go to $E2$. This involves a distance of 8, and the minimum distance from $E2$ to F is 5. Hence the minimum distance from $D1$ via $E2$ to F is $8 + 5 = 13$.
- we can go to $E3$. This involves a distance of 9, and the minimum distance from $E3$ to F is 5. Hence the minimum distance from $D1$ via $E3$ to F is $9 + 5 = 14$.

Thus the optimal policy at $D1$ is to go to $E2$ (so that the sign-post at $D1$ is towards $E2$), and the shortest path from $D1$ to F is 13.

Now repeat this procedure for nodes D2 and D3. You should find that:

- the optimal policy at $D2$ is to go either to $E1$ or $E3$ (both of these choices are equally good) and the shortest path to F is 8.
- the optimal policy at $D3$ is to go to $E3$ and the shortest path to F is 9.

We now move to layer C and find the optimal policy at nodes $C1$, $C2$ and $C3$. And then the same procedure with $B1$, $B2$, and $B3$, and so on until we find the optimal policy at (i.e. the shortest route from) point A.

The procedure above implemented the *Backward (Pass) DP* algorithm. We can divide it into the following steps.

- Divide the problem into stages, numbered from the end: (5) point A to layer B; (4) layer B to layer C; (3) layer C to layer D; (2) layer D to layer E; (1) layer E to point F.
- Start at the penultimate stage of the problem, which is stage (2) in our example;

- *Repeat:*

  (1) find the optimal policy from each state of the current stage to the ultimate destination, making use of our knowledge of the shortest route for later stages which we have already worked out;

  (2) record the optimal policy and the minimum cost at each state;

  (3) go back one stage.

  Until you get to the last stage, at which point the problem is solved.

12.2 General Dynamic Programming Problems

Dynamic programming is appropriate for problems that can be expressed in the same form as the example above. The characteristics are:

- An objective function must be maximised or minimised.
- Decisions are taken in sequence.
- The problem can be divided into *stages*.
- Each stage has a number of possible *states*.
- An *action* (decision) transforms the current state to a state in the next stage. That is, an action tells us how best to move from the current state to a state in the next stage.
- Rewards or penalties are associated with state transformations.

A crucial assumption is that how a state was reached is irrelevant to future decisions. In the stagecoach example, this means that the shortest path from a certain state to F is the same, irrespective of how we arrived at this state while coming from A. This is also called the *Markov property*, the key property of Markov chains.

We now introduce a mathematical formalism to systematically solve a DP. It is convenient to have a systematic way of expressing these problems.

1) label the stages and states associated with the problem.

Let n be the number of stages remaining at any time. This goes backward in time since our Backward DP algorithm goes backward in time.

Let (n, s) denote state s of stage n. For example, state $C2$ is referred to as state $(3, 2)$, as in the diagram below. The set of all states is called the state space. Here we take it to be $\{1, 2, 3\}$ although it could vary with the stage, for example if there was an additional state $D4$.

This yields the following form for the Stagecoach example:

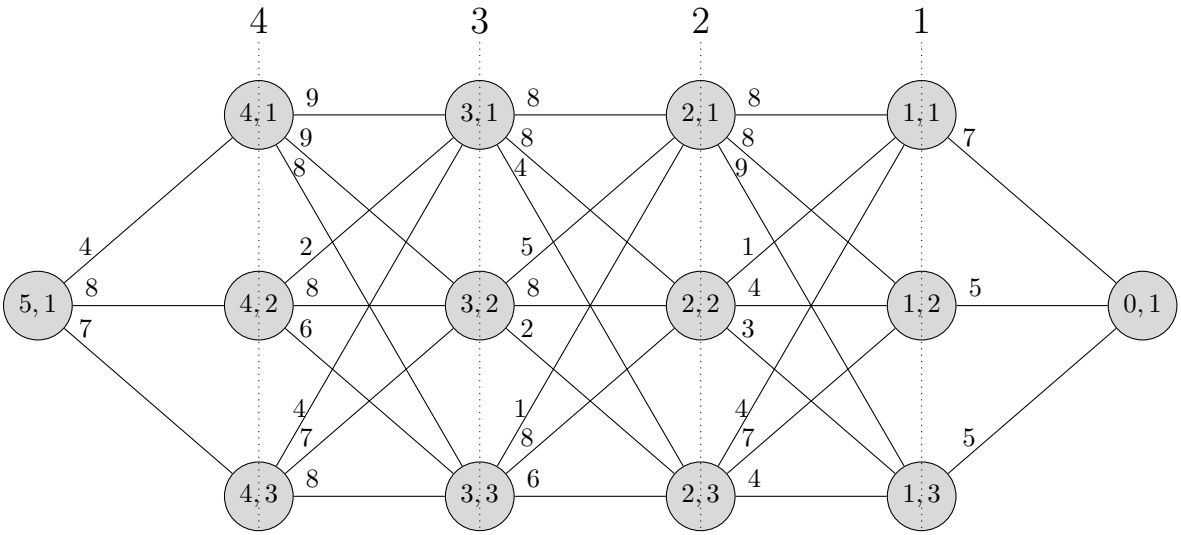


Figure 12.2: Numbering the states in the Stagecoach problem

2) label the possible actions (decisions), and the consequences of those actions (i.e. the transitions).

We call the set of possible actions the action space. The way we label an action/decision depends on the problem. For the shortest path example above, it is most convenient to label the decisions as the states we want to get to at the next stage. Thus the possible actions/decisions are 1,2,3 and so the action space is $\{1, 2, 3\}$.

Let x be a decision variable, so that $x = 1$ means go to state 1. A *decision rule* is a rule (i.e. function) that takes as input the state we are in and returns a decision (i.e. an action). Hence $x(3, 1) = 3$ means that we are at point $(3, 1)$ and choose to go toward point $(2, 3)$. Note that we use the same notation for a decision variable and a decision rule.

Let $t(n, s, x)$ be the *transition function*. This specifies the state that we reach at the next stage as a consequence of our decision. In the simplest case, we just have $t(n, s, x) = x$: e.g. $t(3, 1, 3) = 3$, i.e. if we are at point $(3, 1)$ and we choose decision 3 we will end up at state 3 of the next stage, i.e. point $(2, 3)$.

This seems to be just an unnecessary formalism at this point. The "distribution effort" example of Section 12.3 will show that it might be advantageous to differentiate between actions and states. This becomes very important in stochastic dynamic programming (next lecture), when we cannot fully control where we arrive in the next stage, and hence a clear distinction between actions and states must be drawn.

3) specify the cost (or reward), c, associated with moving to the next stage.

We describe the cost function $c(n, s, x)$ as a function of the current state and the state

where we are moving to. That is, if we are currently in state s of stage n, and we move into state x in state $n-1$, then we incur a cost $c(n, s, x)$. In the stagecoach example, $c(2, 1, 3) = 9$, as the cost of the arc from $(2, 1)$ to $(1, 3)$ is 9.

In the stagecoach example, we wish to minimise the total cost. In other applications, the function c may represent a reward incurred during this transition, in which case we wish to maximise the total reward.

4) provide a recursive formula that identifies the optimal policy for states in stage n, provided the optimal policies in stage $n-1$.

Let $f(n, s)$ denote the value of state s at stage n. That is, the total cost (or reward) from that state onwards (including that state) to the end state if the optimal policy is followed. Hence, for the Stagecoach example above, $f(2, 1) = 13$, since the shortest path from state 1 of stage 2 (node (2,1)) to the final stage has cost 13.

In the simplest scenario, we can compute the overall cost (or reward) associated with a given decision, by adding the following two terms:

(1) The immediate cost (or reward) c in the present stage associated with that decision;

(2) The value of the state that will be reached via this decision.

The optimal decision is the decision that minimises (if we are working with costs) or maximises (if we are working with rewards) this sum. Thus for a minimisation problem (e.g. the shortest path example), we have

$$f(n, s) = \min_x \left[c(n, s, x) + f(n-1, t(n, s, x))\right],$$

whereas for a maximisation problem we have

$$f(n, s) = \max_x \left[c(n, s, x) + f(n-1, t(n, s, x))\right].$$

Here x ranges over all possible decisions: that is, all states at stage $n-1$ that can be reached from state (n, s).

The above formulas provide *Bellman's functional recursion equations* for this scenario.

It is not always possible to decompose the cost/reward into these two terms. What matters is that provided all values $f(n-1, s)$, we are able to efficiently compute the values $f(n, s, x)$. Here $f(n, s, x)$ denotes the value of moving from state (n, s) to state $(n-1, t(n, s, x))$, and using the optimal policy from then onwards. Using this general notation, the equations give

$$f(n, s) = \min_x f(n-1, x, s),$$

for minimisation problems, and

$$f(n, s) = \max_x f(n-1, x, s),$$

for maximisation problems. Using this notation, we had $f(n, s, x) = c(n, s, x) + f(n-1, t(n, s, x))$ in the Stagecoach Problem. A different type of example would be $f(n, s, x) = p(n, s, x)f(n-1, t(n, s, x))$, where $p(n, s, x)$ are given multipliers.

12.2.1 How does dynamic programming save time?

The purpose of DP is to dramatically reduce the number of calculations that would be needed in complete enumeration. The total number of possibilities can grow *exponentially* with the number of states and becomes computationally infeasible very soon.

DP can eliminate most possibilities and thus give an efficient algorithm for large instances. The number of computation steps will grow only *linearly* with the number of states.

To illustrate this, let us compute the total number of elementary computational steps both for complete enumeration and for dynamic programming in the Stagecoach example. We consider elementary arithmetic operations (pairwise additions, subtractions, multiplications, divisions) and comparisons as the relevant computational steps.

It is easy to see that there are $3^4 = 81$ paths between A and F. For each of the 81 paths, we need to perform 4 addition operations to compute the corresponding cost (for example, for path $A \to B2 \to C1 \to D3 \to E2 \to F$ we will need to compute cost by taking the sum $8 + 2 + 4 + 7 + 5 = 26$). This means that, using complete enumeration, the total number of pair-wise additions is $81 \cdot 4 = 324$. Once computed the costs for the 81 paths, we have to compare them to establish the path of minimum cost. This can be done with 80 comparisons. Indeed, suppose for example that, in whichever order we choose the 81 paths, the first 5 path lengths are 92, 86, 87, 83 and 89. Determining that the one of minimum cost among these 5 is the fourth, with cost 83, can be done as follows:

(1) Compare 92 and 86; the best so far is 86.

(2) Compare 86 and 87; the best remains 86.

(3) Compare 86 and 83; the best becomes 83; and finally

(4) Compare 83 and 89; the best (i.e. minimum) of the five distances is 83.

Thus five distances require 4 $(= 5 - 1)$ pair-wise comparisons, and similarly 81 distances require 80 $(= 81 - 1)$ comparisons. Consequently, the total number of (pair-wise) additions and comparisons using complete enumeration is $324 + 80 = 404$.

Using DP. Let us now use DP to compute the total number of arithmetic operations and comparisons.

- First, going from level E to F in stage 1: we simply have 3 distances to record (7, 5 and 5) and no additions or comparisons.
- Next, from level D to level E in stage 2. For example, from $D1$ the shortest distance is the least of $8 + 7$ via $E1$, $8 + 5$ via $E2$, and $9 + 5$ via $E3$, so that in stage 2 we will have three lots of [3 pair-wise additions and 2 pair-wise comparisons], altogether 9 (pair-wise) additions and 6 (pair-wise) comparisons.
- The same applies for stages 3 and 4: 9 additions and 6 comparisons in both these stages.
- Finally, from A to level B in stage 5, we have 3 additions and 2 comparisons.

Consequently in this example, DP requires 30 additions and 20 comparisons, that is, 50 computational steps altogether. Compared with 404 for complete enumeration, we see that DP is approximately 8 times more efficient than complete enumeration. If the number of states increases, this factor will become much larger. It is a matter of seconds to solve instances with hundreds of thousands of states via DP, whereas the entire lifespan of the universe would still not be enough to perform complete enumeration, even using today's fastest supercomputers.

12.3 Distribution of effort

Let us now consider another common application of dynamic programming, which can be used to find the most efficient distribution of tasks over different periods, or finding the most efficient use of certain resources. This will be illustrated by two examples:

12.3.1 Distributing tasks over time

Example 12.1. An engineering company is contracted to build nine trains during a 3-month period, so that the nine trains can be delivered to the customer at the end of the third month. For whatever reason, the cost of building x trains in month m is $w_m x^2$, where the values for w_m are given in the table below, and w_m is measured in millions of £. E.g., if they build 4 trains in month 2, the total cost then will be $3 \cdot 16 = 48$ (£millions).

Month m	1	2	3
w_m	6	3	2

How many trains should be built in each month to minimise total production costs?

We formulate this as a DP problem. Let

$$\begin{aligned} n &= \text{number of months remaining at the start of any stage (i.e. month);} \\ s &= \text{total number of trains still to be produced at the start of this stage;} \\ x &= \text{number of trains to produce during the current stage;} \\ c(n,s,x) &= \text{cost of producing } x \text{ trains in state } s \text{ at stage } n. \end{aligned}$$

In stage 1 we are at the beginning of month 3: $n = 1$ corresponds to $m = 3$. Similarly, $n = 2$ corresponds to $m = 2$, and $n = 1$ to $m = 3$ (note that the relationship between m and n is given by $m = 4 - n$ for $n = 1, 2, 3$). Consequently, when computing $c(n, s, x)$, we use the multiplier w_m for $m = 4 - n$, that is,

$$c(n, s, x) = w_{4-n} x^2$$

In this example, the natural transition function to use is $t(n, s, x) = s - x$: if there are s trains still to be produced, and we produce x new ones in the current stage, then $s - x$ will be left for all subsequent stages.

By the end of month 3, the entire demand has to be satisfied. Hence from state $(1, s)$, the only allowed action is $x = s$: in month three, the company must produce whatever it takes

to meet the full demand. For all other stages, the possible actions available in state s are all x such that $0 \leq x \leq s$. We impose $x \leq s$ as the company should not build more than the required number of trains.

In month 1, we have the entire demand 9 to satisfy. Hence for $n = 3$ there is only one state, namely, $(3, 9)$.

The Bellman equation gives

$$f(n, s) = \min_x [w_{4-n}x^2 + f(n-1, s-x)],$$

where the set of actions is $0 \leq x \leq s$ for $n = 1, 2$ and $x = s$ for $n = 3$. The optimal solution to the problem will be given by the variable $f(3, 9)$.

Let $x^*(n, s)$ denote the optimal production at stage n in state s.

Stage 1 $(n = 1)$**, month 3**

This is month 3, with $x = s$ being the only possible action. Hence $x^* = s$, and $f(1, s) = w_3 s^2 = 2s^2$. Thus we obtain the table below.

s	$x^*(1, s)$	$f(1, s)$
0	0	0
1	1	2
2	2	8
3	3	18
4	4	32
5	5	50
6	6	72
7	7	98
8	8	128
9	9	162

Stage 2 $(n = 2)$**, month 2**

Now $w_2 = 3$, and hence $f(2, s) = \min\{3x^2 + f(1, s-x) : 0 \leq x \leq s\}$.

Working through all the options and choosing the optimum x for each s yields:

$s\backslash x$	0	1	2	3	4	5	6	7	8	9	$x^*(2, s)$	$f(2, s)$
0	**0**										0	0
1	**2**	3									0	2
2	8	**5**	12								1	5
3	18	**11**	14	27							1	11
4	32	21	**20**	29	48						2	20
5	50	35	**30**	35	50	75					2	30
6	72	53	**44**	45	56	77	108				2	44
7	98	75	62	**59**	66	83	110	147			3	59
8	128	101	84	**77**	80	93	116	149	192		3	77
9	162	131	110	99	**98**	107	126	155	194	243	4	98

E.g. when $s = 3$, the formula becomes $f(2,3) = \min\{3x^2 + f(1,3-x)\}$, and x can be 0, 1, 2 or 3. Substituting these values, we get:

$$\begin{aligned} f(2,3) &= \min\{3\cdot 0^2 + f(1,3),\ 3\cdot 1^2 + f(1,2),\ 3\cdot 2^2 + f(1,1),\ 3\cdot 3^2 + f(1,0)\} \\ &= \min\{18, 11, 14, 27\} = 11. \end{aligned}$$

These are the values in row 3. In each row, the optimal value is **bold**; these provide the x^* and $f(2,s)$ values.

Stage 3 $(n = 3)$**, month 3**

This is the start of the production process, and we only have a single state $s = 9$. We use $w_3 = 6$, and thus we have to compute $f(3,9) = \min\{6x^2 + f(2,9-x)\}$. Working through all the possible values of x from 0 to 9 and choosing the optimum yields $x^*(3,9) = 1$ or 2. These two choices both give $f(3,9) = 83$.

$s\backslash x$	0	1	2	3	4	5	6	7	8	9	x^*	$f(2,s)$
9	98	**83**	**83**	98	126	170	227	299	386	486	1,2	83

Evaluating the optimal policy

We now work forward to evaluate the outcome of the optimal policy. We obtain two possible solutions for $x^*(3,9) = 1$ and for $x^*(3,9) = 2$.

- $x_3 = x^*(3,9) = 1$ moves to state $(2,8)$, giving $x_2 = x^*(2,8) = 3$. Then we move to state $(3,5)$, with optimal policy $x_1^* = 5$. Hence we produce 1 train in month 1, 3 trains in month 2, and 5 trains in month 3.
- $x_3 = x^*(3,9) = 2$ moves to state $(2,7)$, giving $x_2 = x^*(2,7) = 3$. Then we move to state $(3,4)$, with optimal policy $x_1^* = 4$. Hence we produce 2 trains in month 1, 3 trains in month 2, and 4 trains in month 3.

Possibly a secondary objective might be applied in order to choose between the two possible production schedules, and a sensible one would be to minimise the variability from month to month, so that the second schedule above would then be chosen.

12.3.2 Distributing resources to groups

We now give another example of distribution of effort is given. This example is from Sec 11.3 of the Hillier-Lieberman book (ed. 7).

Example 12.2. ESA, the European Space Agency, is conducting research to solve a certain engineering problem needed for a Mars mission. Three research teams (1, 2, and 3) are currently working on the project, using different approaches. The problem is solved if at least one of them succeed. It is estimated that under the current circumstances, the three teams will fail with probabilities 0.4, 0.6, and 0.8, respectively. Hence the current probability of failure is $0.4 \cdot 0.6 \cdot 0.8 = 0.192$.

ESA has therefore mandated two more top scientists to these projects. The next table shows the probabilities that each of the teams fail if 0, 1, or 2 more scientists are added.

(Columns correspond to the teams, and rows to the number of new scientists.) Determine how the two scientists should be allocated to minimise the probability that all three teams will fail.

	1	2	3
0	0.4	0.6	0.8
1	0.2	0.4	0.5
2	0.15	0.2	0.3

Although the question involves probabilities, it is still a deterministic dynamic programming question. Our aim is to decide the numbers x_1, x_2, and x_3 of scientists to assign to projects 1, 2, and 3, respectively, under the constraint $x_1+x_2+x_3 = 2$, such that the product

$$p_1(x_1)p_2(x_2)p_3(x_3)$$

is minimised, where $p_i(x_i)$ is the failure probability of team i if x_i additional researchers are added.

Let us formulate this as a dynamic programming problem. Let us first assign scientists to team 3, then to team 2, then to team 1 (this could be done in an arbitrary order). Hence stage $n = 1, 2, 3$ corresponds to assigning scientists to team n. The state (n, s) represents the number of scientists s remaining at this stage, and the action x in state (n, s) is the number of scientists assigned to team n. Clearly, in state (n, s) the only actions available are $x \leq s$ because we cannot assign more scientists than available.

The natural transition function is $t(n, s, x) = s - x$: if we allocate x scientists at stage n, then $s - x$ remain for the further stages.

The cost function $f(n, s)$ in state (n, s) is the minimum probability that all teams $1, \ldots, n$ fail, if we have s scientists available to assign to them. For stage $n = 1$,

$$f(1, s) = \min_{x \leq s} p_1(x)$$

for $s = 0, 1, 2$. Bellman's equation for stages $n = 2, 3$ gives

$$f(n, s) = \min_{x \leq s} p_n(x) f(n-1, s-x)$$

for $s = 0, 1, 2$. This is remarkably different from the previous examples: we do not have an additive cost $c(n, s, x)$, but we multiply the probability $p_n(x)$ of team n failing if we assign x researchers to it, with the optimum value $f(n-1, s-x)$ from the previous stage. Still, this is an appropriate functional recursion equation, as it enables us to compute $f(n, s)$ from the values at stage $n - 1$.

Let $x^*(n, s)$ denote the optimal action at stage n in state s.

Stage 1 $(n = 1)$

For team 1, we could in theory allocate $x < s$ scientists. This would make sense if adding more scientists could increase the failure rate. This is not the case according to the table: the optimal policy is to assign the maximum number available.

s	$x^*(1,s)$	$f(1,s)$
0	0	0.4
1	1	0.2
2	2	0.15

Stage 2 ($n = 2$)

We now have $f(2,s) = \min\{p_2(x)f(1,s-x) : 0 \le x \le s\}$. We have all possible options listed below:

$s\backslash x$	0	1	2	$x^*(2,s)$	$f(2,s)$
0	**0.24**			0	0.24
1	**0.12**	0.16		0	0.12
2	0.09	**0.08**	**0.08**	1,2	0.08

Stage 3 ($n = 3$)

As we start the assignment with team 3, we have a single state $s = 2$: all scientists are still available. The formula is $f(3,2) = \min\{p_3(x)f(2,2-x)$.

$s\backslash x$	0	1	2	x^*	$f(3,s)$
2	0.064	**0.06**	0.072	1	0.06

Evaluating the optimal policy

The value $f(3,s) = 0.06 = 6\%$ gives the answer: this is the lowest joint failure probability that could be achieved by assigning additional scientists. To evaluate the optimal policy, we first see that $x^*(3,2) = 1$: we should assign one scientists to team 3. This moves to state $(2,1)$. For state $(2,1)$, the optimal policy is $x^*(2,1) = 0$. This moves to state $(1,1)$, with $x^*(1,1) = 1$.

Hence we have a single optimal solution: one scientist should be assigned to team 1, and one to team 3.

12.4 Stochastic Dynamic Programming

We now demonstrate how dynamic programming can be applied to settings involving uncertainty. Let us revisit the stagecoach example with the network in Figure 12.2. We want to get to the point labelled (0,1) from the point labelled (5,1), minimising the distance travelled.

But we will now assume that the coach sometimes takes the wrong turns, due to poor maps of the wild west. More precisely, at all the nodes of stages $2,3,4,5$, the probability that we traverse the link that we decided to take will be 0.5, whereas we will traverse the two wrong links with probability of 0.25 each.

For example, if you were at $(2,3)$ and your preferred link was to $(1,3)$, there would be a 50% chance that you would take the link to $(1,3)$, a 25% chance that you would do a mistake and take the link to $(1,2)$, and a 25% chance that you would do a mistake and take the link to $(1,1)$. Of course, from $(1,2)$ you are certain to get to $(0,1)$.

Hence the expected distance to be covered from $(2,3)$ to $(0,1)$ will not be 9 via $(1,3)$ as in the original deterministic problem, but $0.5 \cdot (4+5) + 0.25 \cdot (4+7) + 0.25 \cdot (7+5) = 10.25$.

As consequences of the randomness in outcomes, there are two aspects in which this problem differs from the deterministic shortest path.

(i) We can no longer think in terms of simply maximising or minimising an objective function. Instead, with stochastic dynamic programming (SDP) we are normally concerned with maximising or minimising the *expected value* of an objective function.

(ii) The solution is no longer a single optimal route. Rather, the optimal solution is a policy, which to each possible state assigns an action (namely, the preferred link to cross), whose outcome is uncertain due to the intrinsic randomness of the process. However, in expectation the optimal value of the objective is achieved.

Assume that we want to minimise the expected distance through the network. What is the optimal policy at each node? What is the expected distance from $(5, 1)$ to $(0, 1)$?

Begin with the states (nodes) at the last stage of the problem. There is no randomness involved. Thus: from $(1, 1)$ the distance is 7; from $(1, 2)$ the distance is 5; and from $(1, 3)$ the distance is 5.

Now consider the nodes at stage 2. What are the optimal policies from $(2, 1)$, $(2, 2)$, and $(2, 3)$?

From $(2, 1)$ we first compute the expected distances if we were able to follow the 3 possible links (just as in the deterministic setting):

- we can go to $(1, 1)$. This involves a distance of 8, and the distance following the optimal policy from $(1, 1)$ to $(0, 1)$ is 7. Hence the (expected) distance from $(2, 1)$ via $(1, 1)$ to $(0, 1)$ is $8 + 7 = 15$.

- we can go to $(1, 2)$. This involves a distance of 8, and the distance following the optimal policy from $(1, 2)$ to $(0, 1)$ is 5. Hence the expected distance from $(2, 1)$ via $(1, 2)$ to $(0, 1)$ is $8 + 5 = 13$.

- we can go to $(1, 3)$. This involves a distance of 9, and the minimum distance following the optimal policy from $(1, 3)$ to $(0, 1)$ is 5. Hence the expected distance from $(2, 1)$ via $(1, 3)$ to $(0, 1)$ is $9 + 5 = 14$.

Recall that the different actions correspond to intended directions; however, we only get there with probability 50% and end up using the other links with probability 25%.

- *Intended action* $(1, 1)$: the expected distance to $(0, 1)$ is $(0.5 \cdot 15) + (0.25 \cdot 13) + (0.25 \cdot 14) = 14.25$

- *Intended action* $(1, 2)$: the expected distance to $(0, 1)$ is $(0.25 \cdot 15) + (0.5 \cdot 13) + (0.25 \cdot 14) = 13.75$

- *Intended action* $(1, 2)$: the expected distance to $(0, 1)$ is $(0.25 \cdot 15) + (0.25 \cdot 13) + (0.5 \cdot 14) = 14$

Hence the optimal policy at $(2, 1)$ is to attempt to go to $(1, 2)$, and the expected distance to $(0, 1)$ if the optimal policy is followed is 13.75.

So we can similarly find the optimal policy and expected distance for the other states at stage 2. We then work backwards to find the optimal policy and expected distance for all possible states in earlier stages. Etc., etc.

In general, note that the optimal policy for a stochastic shortest route problem may look quite different from that for the deterministic case.

12.4.1 Mathematical Formulation of Stochastic DPs

In the above example, actions corresponded to the states in the next stage. The driver had an intended direction, but with a certain probability he was not able to get there. The model is more general than that: actions can correspond to arbitrary probability distributions on states. For example, another possible action from state $(2, 1)$ could be to follow every link with equal probability.

Consider state s at stage n. Every possible action x corresponds to a probability distribution on the states of stage $n-1$. Formally, given an action x available at a state (s, n), the probability that action x will lead to state j in stage $n-1$ is denoted by $p(n, s, x, j)$. We require for each state (n, s) and each action x available from s that

$$\sum_j p(n, s, x, j) = 1$$

holds, where summation is over all states j at stage $(n-1)$.

We define $f(n, s)$ as the *expected cost (or reward)* from state (n, s) onwards, until the final state is reached.

Assume we have a cost function $c(n, s, x, j)$ giving the cost of taking action x from (n, s), and arriving at state $(n-1, j)$. Then the functional recursion equation for a cost minimisation problem is given as

$$f(n, s) = \min_x \sum_j p(n, s, x, j) \left(c(n, s, x, j) + f(n-1, j)\right).$$

In the stagecoach model, the cost $c(n, s, x, j)$ did not depend on the action x (i.e. where we intended to go), but only on (n, s) and j, that is, where we really ended up. In other models, the cost might be independent on j, and only depend on the action taken.

12.4.2 Stock option example

We have the option of buying a certain amount of stock of a company for £103 during the next 10 days, regardless of the market price of the stock on the day. We have no obligation to exercise this option, but if we do, we can do so in any of the next 10 days. We can exercise the option at most once.

The price on day 1 is £100, and the stock price changes randomly according to the following stochastic model: every day, the price can go up by £1 with 40% probability, can stay the same with 20% probability, and can decrease by £1 with 40% probability. Our aim is to devise an optimal policy for exercising the option: every day, depending on the current price, we need to decide whether to exercise the option to buy the stock, or wait for the next day.

If the price on any date is $s > 103$, then our revenue is $s - 103$ if we use the option. Since on day 1 the price is 100, we are better off not exercising the option on the first day, because doing so would incur a loss of 3.

We use stochastic dynamic programming to model this problem. Our stages correspond to the days, numbered backwards: stage $n = 1$ is day 10, stage $n = 2$ is day 9, etc. Stage $n = 11$ is day 0 - the current day.

The states correspond to the current price: (n, s) means that in stage n, the stock price is s, and we have not yet used the option; there will be a further special state $(n, \star)$. For $n = 1$ (day 10), the price can be between 90 and 110, as it could have increased or decreased by at most 10. Hence we have 21 possible states: $90 \leq s \leq 110$. For $n = 2$ (day 9), the possible states are $91 \leq s \leq 109$, and so on, for $n = 10$ (day 1, tomorrow) the possible states are $s \in \{99, 100, 101\}$. For the final stage, $n = 11$, we have only one state, $s = 100$ – this is the price of the stock today.

The special state $(n, \star)$ means that we have already used the option before. That is, $(2, \star)$ means that we used the option in one of the first 8 days (note that $n = 2$ refers to day 9).

The value $f(n, s)$ will be the expected maximum revenue we can achieve from state (n, s). The value of the special state is always $f(n, \star) = 0$. There are two possible actions in state (n, s): either we buy the stock using the option, or we wait.

- **Buy.** In this case, our revenue is $s - 103$; this can be negative if the current price s is lower than £103. We move from state (n, s) to the special state $(n - 1, \star)$.

- **Wait.** In this case, we will move to one of the states $(n - 1, s - 1)$, $(n - 1, s)$, or $(n - 1, s + 1)$, with respective probabilities 40%, 20%, and 40%. Our expected revenue will be

  $$0.4f(n - 1, s - 1) + 0.2f(n - 1, s) + 0.4f(n - 1, s + 1).$$

  This applies for $n > 1$; for $n = 1$, if we do not use the option on the last day, the revenue will be 0.

The Bellman equations are obtained by taking the maximum of the two values. For $n > 1$, it gives

$$f(n, s) = \max\{s - 103, 0.4f(n - 1, s - 1) + 0.2f(n - 1, s) + 0.4f(n - 1, s + 1)\},$$

and for $n = 1$, we have

$$f(1, s) = \max\{s - 103, 0\}.$$

These formulas can be easily implemented in Excel; you can find the implementation on Moodle. The next table shows the values of $f(n, s)$ rounded to the second decimal place.

$f(n,s)$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n=10$	$n=11$
$s=90$	0										
$s=91$	0	0									
$s=92$	0	0	0								
$s=93$	0	0	0	0							
$s=94$	0	0	0	0	0						
$s=95$	0	0	0	0	0	0					
$s=96$	0	0	0	0	0	0	0				
$s=97$	0	0	0	0	0	0	0	0			
$s=98$	0	0	0	0	0	0	0	0.01	0.02		
$s=99$	0	0	0	0	0	0.01	0.02	0.04	0.05	0.07	
$s=100$	0	0	0	0	0.03	0.05	0.08	0.11	0.14	0.17	0.2
$s=101$	0	0	0	0.06	0.1	0.16	0.2	0.26	0.3	0.35	
$s=102$	0	0	0.16	0.22	0.32	0.39	0.46	0.52	0.58		
$s=103$	0	0.4	0.48	0.62	0.7	0.8	0.87	0.94			
$s=104$	1	1	1.16	1.22	1.32	1.39	1.46				
$s=105$	2	2	2	2.06	2.1	2.16					
$s=106$	3	3	3	3	3.03						
$s=107$	4	4	4	4							
$s=108$	5	5	5								
$s=109$	6	6									
$s=110$	7										

The value of the final state $f(11,100) = 0.202418$ shows that the expected revenue we can get from this option is 20 pence.

To decide whether to buy or wait at a certain state (n,s), we need to compare $s-103$ with $f(n,s)$. If $f(n,s) > s-103$ then we wait, because the expected revenue is higher than what we would make by selling on this day; otherwise we must have $f(n,s) = s-103$, in which case we buy because we earn exactly the optimal expected revenue for state (n,s).

For example, $f(5,106) = 3.0256$. This means that if on day 6 the price is 106, the optimal policy is to wait, as the expected revenue is higher than the 3 we would gain by buying the stock today.

Suggested reading

The Hillier-Lieberman textbook [HL] provides a detailed description of dynamic programming, with many excellent worked examples and many further exercises. In particular, you can find worked examples of stochastic dynamic programming in Section 11.4 (ed 7).

Exercises

Exercise 12.1. Suppose a shortest path problem is to start at A, finish at K, and pass through nine intermediate levels, B, C, D, E, F, G, H, I, and J, each containing seven

individual places (states).

How many pair-wise calculations would be required to identify the shortest path *(a)* by complete enumeration; or *(b)* by DP? How many times more efficient is DP in this case?

Exercise 12.2. Suppose the numbers along the individual links in the stagecoach example (Figure 12.1) represent net revenues in £'00 earned by a travelling salesman going from town A in one county to town F in another county. He traverses four intermediate counties B, C, D, and E, and can stop overnight at one of three towns (e.g. $B1, B2$, or $B3$). The numbers in the network represent the revenues he can make while travelling between two towns. E.g. number 9 between $B1$ and $C1$ means he can make £900 on the way from $B1$ to $C2$. Calculate how he can maximise his total net revenue between A and F.

Exercise 12.3. A firm is running an extended sales campaign in a region consisting of towns A, B, C, and D, each with its own weekly newspaper. The firm has decided to insert seven identical advertisements and wishes to allocate them between the papers. Its representatives in the towns estimate that the sales of the firm's product are likely to be affected by the placing of advertisements as shown in the table below, and the revenue earned (in £) will be as in the table. The columns represent the towns, and the rows the number of advertisements published in this town.

	A	B	C	D
0	2000	4000	3600	6000
1	2400	4500	3700	6100
2	2500	4700	3900	6200
3	2600	4800	4200	6300
4	2700	4800	4300	6400

For example if they decide to place 3 successive adverts in the newspaper of town A (i.e. one in each of three separate weeks during the campaign), two each in B and C, and none in D, the total revenue will be £$\cdot(2600 + 4700 + 3900 + 6000)$ =£17,200.

Use DP to decide how many advertisements the firm should place in each newspaper.

Exercise 12.4. A firm producing units of machinery must deliver 1 unit to customers by the end of May, 2 by the end of June, and 4 by the end of July, i.e. seven units in total between months May and July. They are able to produce units earlier, e.g. 2 in May, 2 in June, and 3 in July would also be a feasible production plan. Units not required for delivery in the month they are produced can be held in stock for later delivery.

The firm can produce a maximum of four units in any month, but the production costs differ as shown below. Costs are in £'000; the rows correspond to the number of units produced in the particular month; there is no stock holding cost. Assume that production costs are zero if no units are produced in any month.

	May	June	July
1	6	4	5
2	9	7	6
3	10	10	10
4	11	13	11

(a) Use DP to produce all the minimum cost production schedules for the firm.

(b) Assume the stock holding cost is not zero, but £500 per unit per month, calculated each month on the number of units held in stock at the end of the month. Write the modified functional recursion equations. (You do not have to solve the problem.)

Exercise 12.5. A voluntary organisation is due to receive a grant of £100,000 in 2016. There are four projects on which it could spend such a grant. The relative success of each project is determined by the resources put into the project.

The table shows the relative success (from 0 to 10) for each combination of project and allocation, ranging from no success at all = 0, up to very successful indeed = 10. The columns represent the allocated sums in £'000, and the rows correspond to the projects.

	0	10	20	30	40	50	60
1	0	2	10	-	-	-	-
2	0	0	3	3	9	9	10
3	0	4	4	6	10	10	-
4	0	3	4	10	-	-	-

The organisation has calculated that the maximum overall success (i. e. the sum of the relative successes of the projects) that it can get out of the grant is by allocating £20,000 to project 1, £40,000 to project 2, £10,000 to project 3, and £30,000 to project 4.

(a) Give the recursive equation which would have solved the above problem. Given the allocation above, what is the maximum overall success?

(b) Now suppose the value of each project to the organisation is not the same but, in discussion with the management committee, weights (i.e benefits) are assigned to each project as follows:

Project	1	2	3	4
Weight	4	5	7	6

For example, if projects 1 and 2 are equally successful, project 2 would be worth 25% more to the organisation than project 1 [i.e. $= 100 \cdot (5-4)/4$]. Define a recursive equation that would take account of these weights and calculate the best allocation.

(c) How much should the voluntary organisation allocate to each project to maximise the overall weighted success?

(d) Compare the results obtained using the two formulations.

(e) If the organisation learns the grant could be cut to either £50,000 or £80,000, but the amounts spent per project would still be multiples of £10,000, what would you advise them to do to maximise the overall weighted success?

Exercise 12.6. Consider a sequence of numbers, such as $(2, 1, 7, 3, 5, 2, 4)$. An increasing subsequence is a subsequence where each number is greater than the previous one. For example, $(2, 7)$, $(2, 3, 5)$, or $(1, 3, 5)$ are all increasing subsequences of the sequence above. In contrast, the subsequence $(2, 7, 3, 4)$ is not increasing, since 7 is followed by 3. In the example, it is easy to see that there is no increasing subsequence of length 4.

Formulate the problem of finding the longest increasing subsequence of a sequence as a dynamic programming problem, and perform the method on the sequence above.

Hint: let $f(n, s)$ denote the length of the longest subsequence that only uses some of the first n numbers, and its last element is less than or equal to s.

Exercise 12.7. A gambler starts with 2 chips. There are to be 3 plays of a game of chance. At each play he can stake any number of chips from zero up to the number he currently has. He then wins or loses the number staked. He needs to end with at least 4 chips to pay the hotel bill.

Give a formulation of the problem which remains valid for N plays, starting with S chips and finishing with at least r chips, where the probability of winning on any play is 0.6. For the numerical example above, devise a complete optimal strategy.

Part II

Selected Topics in Operational Research

Chapter 13

Game theory

13.1 Basic concepts

The theory of games deals with conflict situations in which the outcome depends on the strategies chosen by each of the players. A game is a social situation in which there are several individuals, each pursuing their own interest and in which no single individual can determine the outcome, e.g. many economic situations, warfare. Many conflicts involve an element of chance (e.g. card games, the weather in an agricultural situation). We will not discuss games of pure chance, e.g. roulette, but only games in which the skill and intelligence of the players are useful, that is, games of strategy.

In a game situation there are:

(1) n decision makers (players), where $n \geq 2$.

(2) Rules specifying available courses of action (strategies), known to the players.

(3) A well-defined set of *outcomes*, and rules specifying which outcome will occur.

(4) A *payoff* to each player associated with each outcome, known to all the players.

13.1.1 Classifications of game situations

Games may be classified according to the following characteristics:

(1) *Number of players.* We distinguish *2-person games* and *more-than-2-person games.* In this course, we shall be concerned with 2-person games; that is, the people participating necessarily fall into 2 mutually exclusive groups, each group having identical objectives.

(2) *Nature of payoff.* We distinguish *zero-sum games* from *non-zero-sum games.* A zero-sum 2-person game is one in which for any outcome of the game, one player's gain is the other player's loss. Equivalent to zero-sum games are *constant-sum games*, in which the payoffs to players 1 and 2 sum to a quantity which is constant over the outcomes. In either case, the interests of the two players are directly opposed. These two lectures will deal with zero-sum games.

(3) *Communication between players.* We distinguish *cooperative games* and *non-cooperative games*. In non-zero-sum games, there may be mutual benefits to be obtained from the coordination of strategy choices, with or without side payments. We shall be concerned with non-cooperative games in which no pre-play communications between players is possible.

13.2 Extensive and Normal Forms

An n-person game in *extensive form* is a tree representing the moves of the game. We use the following example to demonstrate the extensive form.

Example 13.1. Simplified poker. There are only two cards involved - an Ace and a Two - and only two players, called Alice and Bob. Each puts £1 in the pot and Alice deals Bob one card, which Bob then looks at. If it is the Ace, Bob must say 'Ace', but if it is the Two, Bob can say 'Ace' or 'Two'.

- If Bob says 'Two', he loses the game and his £1 in the pot.
- If Bob says 'Ace', no matter what the card is, he must put another £1 in the pot. In this case, Alice can either believe him ('fold'), and so lose her £1 in the pot, or she can demand to see the card ('call'). In this case, Alice has to put another £1 into the pot, and then the card is shown. If Bob had the Ace, he wins the pot and so takes £2 from Alice; but if Bob has the Two, Alice wins the pot and so takes £2 from Bob.

This game involves the elements of 'bluffing' and 'calling' involved in real poker, but not surprisingly has not yet taken Las Vegas by storm. The extensive form for the above game is shown in Figure 13.1. It has the following properties:

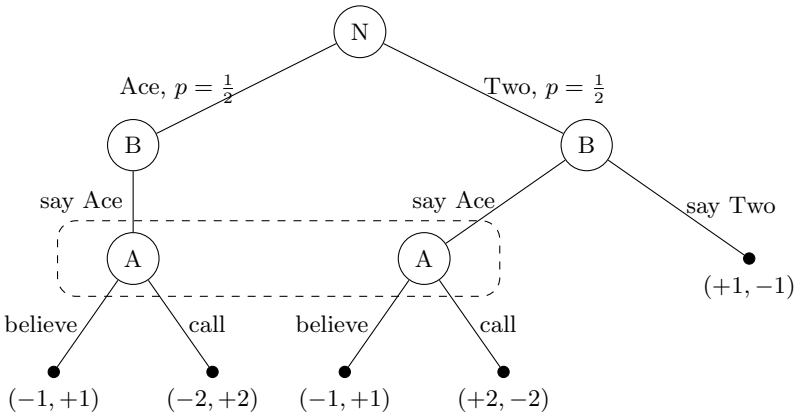


Figure 13.1: The extensive form of Simplified poker

(i) The tree has an initial node.

(ii) Each node is labelled either with N (for Nature) or with the name of one of the n players.

(iii) A node labelled N represents random variables that are important to the game. The edges that lead out of an N node represent the different outcomes of the random variable. Thus, each of these edges is labelled with the outcome and the probability of the outcome.

(iv) The nodes that are labelled with the player names are decision nodes and each edge leading from them represents an action (strategy) of the player.

(v) Each path from the initial node through a leaf describes a scenario of the game; it specifies the outcome of the random variables and the decisions of the players. For each scenario, we can calculate the *outcome of the game* and thus calculate the payoff to each player. We label each leaf with the payoff to the players. In a zero sum game we only need to declare the payoff to the first player (since the other player gets the negative of that).

(vi) The nodes of each player are partitioned into disjoint subsets called *information sets*. Nodes are in the same information set if the player has the same information on each of these nodes. A player is presumed to know which information set he or she is in, but not which node of the set. Any two nodes in the same set must have the same set of choices (edges with identical labels) leading from them.

In the diagram, the two nodes of Alice belong to the same information set (joined in a dotted rectangle). The two nodes of Bob belong to different information sets; depending on the card he sees, Bob knows which of the two nodes he is currently in.

If all the information sets consist of one node only, the game is one of *perfect information*. A finite game of perfect information can be solved (i.e. optimal actions can be obtained) by "reducing it backward", in a method similar to dynamic programming, one of our LT topics.

13.2.1 Normal form of games

An n-person game in strategic (or normal) form consists of a set X_i of strategies for each player i, and a set P_i of payoff functions. If player i chooses $s_i \in X_i$ for $i = 1, \ldots, n$, then the payoff to player j is $P_j(s_1, \ldots, s_n)$. We can imagine that the game is played by each player selecting a strategy and handing it to a referee. He then announces all the strategies and the resulting payoffs to each player.

We shall mainly be concerned with two-person games in strategic form. Player 1 can be thought of as choosing a row of a rectangular array, while player 2 selects a column. In each cell of the array are two numbers giving the resulting payoffs to the two players. A special case arises when one player's gain always equals the other player's loss: the sum of the two entries in each cell is then zero. This is the case in two-player zero-sum games.

We can write the Simplified poker game in strategic form as follows. Both players have two possible strategies:

Alice	A1	believe Bob when he says 'Ace' *(fold)*;
	A2	do not believe Bob when he says 'Ace' *(call)*;
Bob	B1	say 'Two' when you have a Two;
	B2	say 'Ace' when you have a Two *(bluff)*.

Each of these strategies perfectly determines the decisions to be made by the players. We can construct the payoff matrix, containing the expected payoffs for Alice and Bob, with negative values expressing loss. Note that the two numbers always sum up to 0 - this is a zero-sum game. We first present the payoff matrix, and then explain how the entries were obtained.

	B1	B2
A1	$(0, 0)$	$(-1, 1)$
A2	$(-\frac{1}{2}, \frac{1}{2})$	$(0, 0)$

The rules of the game include a probabilistic event, namely the card Bob receives can be an Ace or a Two, both with probability 1/2. The payoffs here were calculated as the *expected values* with respect to this event, as follows.

A1 vs B1 If Bob gets an Ace, he says 'Ace' and Alice believes him (according to strategy A1), so Alice has payoff -1. If Bob gets the Two, he says (according to strategy B1) 'Two', and so Alice gets payoff $+1$ straightaway. Both cards appear with a chance 1/2, so the expected payoff is

$$\frac{1}{2} \cdot (-1) + \frac{1}{2} \cdot 1 = 0.$$

A1 vs B2 No matter what Bob gets, he always says 'Ace'; Alice always believes him and gets payoff -1.

A2 vs B1 If Bob gets an Ace, he says 'Ace', but Alice does not believe him, and hence Alice gets -2. If Bob gets a Two, he says so, and Alice gets $+1$. The expected payoff is

$$\frac{1}{2} \cdot (-2) + \frac{1}{2} \cdot 1 = -\frac{1}{2}.$$

A2 vs B2 Bob always says 'Ace', and Alice never believes him. When the card is shown, if it is an Ace, then Alice gets -2; if it is a Two, then she gets $+2$. The expected payoff is

$$\frac{1}{2} \cdot (-2) + \frac{1}{2} \cdot (+2) = 0.$$

13.3 Two player zero-sum games

Provided a gain to player A is always an equivalent loss to player B (that is, the game is zero-sum), the game can be represented by a payoff matrix where the rows represent strategies

of A, and the columns strategies of B. The payoff is conventionally in terms of gain to the row-player, but may be negative, if it represents a loss.

$$
\begin{array}{cc} & \text{B} \\ & \begin{array}{cccc} 1 & 2 & \dots & m \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \\ \vdots \\ n \end{array} & \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1m} \\ r_{21} & r_{22} & & r_{2m} \\ & & \ddots & \\ r_{n1} & r_{n2} & & r_{nm} \end{bmatrix} \end{array}
$$

In a game, decisions are taken under uncertainty - no probabilities can be ascribed to the occurrence of different opposing strategies. Since one's opponent is assumed rational and has opposing interests, it is reasonable to expect the worst result which your strategy permits.

This leads to the *minimax criterion*:

> **Minimax criterion.** Choose your strategy so as to minimise the maximum loss you can sustain (or equivalently, maximise your minimum gain).

13.3.1 Pure strategy solutions

The minimax criterion may be applied by listing the row minima and column maxima, and finding the largest of the former and the smallest of the latter - in the example these are starred. The largest row minimum is the payoff A can guarantee himself. The smallest column maximum is a loss B can be sure not to exceed. When they are equal (as in the example) the game has *a solution in pure strategies*, and their joint value is the *value of the game* to A. A game is *fair* if its value is zero.

$$
\begin{array}{cc} & \text{B} \\ & \begin{array}{cc} 1 & 2 \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \\ \\ \end{array} & \begin{array}{cc} \begin{bmatrix} 1 & 5 \\ 2 & 3 \end{bmatrix} & \begin{array}{c} \mathbf{1} \\ \mathbf{2}* \\ \\ \end{array} \\ \begin{array}{cc} \mathbf{2}* & \mathbf{5} \end{array} & \end{array} \end{array}
$$

We say that a pair of strategies is *a saddle point* if the payoff for this pair is a maximal element in its column, and at the same time, a minimal element in its row. If either player departs from the saddle-point strategy, he is bound to be worse off. It is easy to verify that if a game has a solution in pure strategies, such a pair of strategies is always a saddle point, and vice versa: every saddle-point corresponds to a solution in pure strategies. As we will

see later, not every game has a saddle point; if it has, there can be more than one. In this case, all saddle-points must have the same value (see Exercise 13.1).

Let us see some further examples. The following game has a solution in pure strategies and is fair.

$$\begin{array}{cc} & \text{B} \\ & \begin{array}{cc} 1 & 2 \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc} 0 & -2 \\ 0 & 2 \end{array}\right] \begin{array}{c} \mathbf{-2} \\ \mathbf{0}* \end{array} \\ & \begin{array}{cc} \mathbf{0}* & \mathbf{2} \end{array} \end{array}$$

The next one has a solution in pure strategies but is not fair.

$$\begin{array}{cc} & \text{B} \\ & \begin{array}{cc} 1 & 2 \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc} 5 & 2 \\ -7 & -4 \end{array}\right] \begin{array}{c} \mathbf{2}* \\ \mathbf{-7} \end{array} \\ & \begin{array}{cc} \mathbf{5} & \mathbf{2}* \end{array} \end{array}$$

The next game does not have a solution in pure strategies: the maximum of row minima is 1, whereas the minimum of the column maxima is 2. The pair of strategies (A1,B1) is not a saddle point: A could change his strategy to A2 and get a better payoff.

$$\begin{array}{cc} & \text{B} \\ & \begin{array}{cc} 1 & 2 \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc} 1 & 5 \\ 2 & 0 \end{array}\right] \begin{array}{c} \mathbf{1}* \\ \mathbf{0} \end{array} \\ & \begin{array}{cc} \mathbf{2}* & \mathbf{5} \end{array} \end{array}$$

The same method applies to larger games. The following 4×4 game has a solution in

pure strategies. The optimal strategies are A3 and B2, and the value of the game is 3.

		B				
		1	2	3	4	
A	1	7	2	5	1	**1**
	2	2	2	3	4	**2**
	3	5	3	4	4	**3**∗
	4	3	2	1	6	**1**
		7	**3**∗	**5**	**6**	

13.3.2 Dominance

If every value in a row of the payoff matrix, R1, say, is greater than the corresponding values of another row, R2, the former strategy *strictly dominates* the second one. Hence, whatever player B's chosen strategy may be, player A will always be better off playing the strategy corresponding to R1 than if he follows the strategy of R2. This is almost the situation with A3 and A2 above, except that the payoff to A is 4 if B plays B4 in both cases. When every value in a row R1 is either greater than or equal to the corresponding value of another row R2, the former *weakly dominates* the second (but not strictly so). We say that a row dominates another if there is either strict or weak domination. Thus, A3 dominates A2, so A should never use A2 and we can remove it from consideration (effectively deleting it from the game).

Similarly, Player B always wishes to pay out as little as possible, so if every value of a column C1 is strictly less than the corresponding value of another column C2, the first column strictly dominates the second, and player B should not choose this second strategy. In an analogous way to player A above, weak dominance is defined for the column player. In the above example, the second column B2 weakly dominates the first column B1.

If we delete strategies of the two players that are weakly dominated, then the value of the game is still unaltered by deletions, but some saddle-points may be eliminated.

From the above matrix, we obtain by domination the following 3×3 matrix.

		B			
		2	3	4	
A	1	2	5	1	**1**
	3	3	4	4	**3**∗
	4	2	1	6	**1**
		3∗	**5**	**6**	

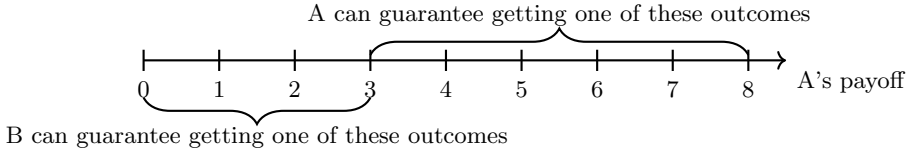
13.4 Mixed strategies

Consider the two-person zero-sum game represented by the payoff matrix below.

		B			
		1	2	3	
A	1	7	2	8	**2**
	2	5	3	4	**3**∗
	3	0	1	6	**0**
		7	**3**∗	**8**	

The row minima have a maximum of 3 and the column maxima have a minimum of 3. Player A can guarantee 3 by playing strategy A2, and Player B can hold A to 3 by playing B2. The central '3' is a saddle point of the matrix. There are two good reasons for claiming that the strategy pair $(2, 2)$ is a solution to this game:

(a) What A can guarantee equals what B can hold him to. Since the game is zero-sum, this outcome must be the solution: A will not accept less than he can guarantee, nor will B let him get more than he can hold him to. Diagrammatically, the possible outcomes (payoffs to A) are the following:

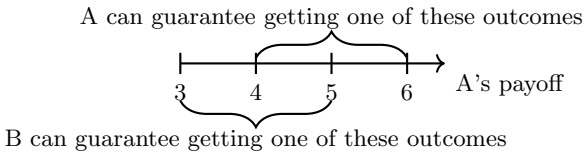


(b) The strategy pair $(2, 2)$ is a saddle point, and therefore an *equilibrium*. This means that neither player has any incentive to change his strategy since this point is the maximum payoff in its column (thus A doesn't want to move) and it is the minimum in its row (thus B doesn't want to move).

Now consider a game matrix with no saddle point.

		B		
		1	2	
A	1	3	6	**3**
	2	5	4	**4**∗
		5∗	**6**	

In this game, the maximum of the row minima is 4, whilst the minimum of the column maxima is 5. We get the following diagram:



No pair of *pure* strategies can be considered to be a solution. Is there any way that A can guarantee more than 4 and B hold him to less than 5? There is, if they randomise their strategy choices. A *mixed strategy* is a combination of pure strategies in a certain proportion, the choice of which strategy to play being made by a random device with the appropriate probabilities.

For example, if A plays 1 with probability $\frac{1}{3}$ and 2 with probability $\frac{2}{3}$, then:

- if B plays 1, the expected gain to A is $\frac{1}{3} \cdot 3 + \frac{2}{3} \cdot 5 = 4\frac{1}{3}$;
- if B plays 2, the expected gain to A is $\frac{1}{3} \cdot 6 + \frac{2}{3} \cdot 4 = 4\frac{1}{3}$.

Hence the mixed strategy $(\frac{1}{3}, \frac{2}{3})$ guarantees A at least $4\frac{1}{3}$ in expectation. It turns out that the best mixture for A is $(\frac{1}{4}, \frac{3}{4})$, which guarantees him $4\frac{1}{2}$ (in expectation). The mixed strategy $(\frac{1}{2}, \frac{1}{2})$ guarantees B a loss of no more that $4\frac{1}{2}$ (in expectation). If A plays $(\frac{1}{4}, \frac{3}{4})$ and B plays $(\frac{1}{2}, \frac{1}{2})$, then neither player has any incentive to change his strategy. Hence by allowing mixed strategies, we regain the two properties (a) and (b) that we noted for the pure strategy solution to the first game.

This is always true for two person zero-sum games. What A can guarantee (in expectation) by choice of the best mixed strategy will always equal what B can hold him to. This is called the *value* of the game. Mixed strategies that guarantee the value of the game in expectation are called *optimal*. Any pair of optimal mixed strategies for A and B will be in equilibrium, and will give both players their guarantees (in expectation).

In a 2×2 zero-sum game, it is possible to write formulas to calculate the optimal mixed strategies explicitly. The existence of such strategies is true for an arbitrary (finite) number of strategies of the player. Von Neumann's famous theorem shows that this holds in arbitrary two-person zero-sum games.

Theorem 13.2 (John von Neumann, 1944)**.** *In every two-person zero-sum game, the maximum expected value the row player A can guarantee equals the minimum expected value the column player B can guarantee.*

In the next section, we will derive this theorem using Linear Programming. Before that, let us remark for the interested reader that the theorem is true under much more general circumstances. In a finite n-person game, we consider a mixed strategy for every player. We say that these mixed strategies form a *Nash equilibrium* if no player can achieve a better expected payoff by unilaterally changing his strategy, that is, if all other players' strategies remain unchanged.

Theorem 13.3 (John Nash, 1951)**.** *In every game with a finite number of players, where every player has a finite number of strategies, there exists a Nash equilibrium.*

13.5 Games and Linear Programming

We derive von Neumann's theorem, by showing that optimal mixed strategies of both players can be obtained using Linear Programming in every two-person zero-sum game. We demonstrate the method on a particular example.

Consider the 3×3 game below; there is no pure strategy solution, and the game is irreducible by dominance. Let (x_1, x_2, x_3) express the mixed strategy of row player A; that is, x_i is the probability of playing strategy i, for $i = 1, 2, 3$. We require $x_1 + x_2 + x_3 = 1$ in order to have a probability distribution. Similarly, let (y_1, y_2, y_3) express the mixed strategy of the column player B.

$$
\begin{array}{cc} & \text{B} \\ & \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc} 1 & 2 & -1 \\ -2 & 1 & 1 \\ 2 & 0 & 1 \end{array}\right] \begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array} \\ & \begin{array}{ccc} y_1 & y_2 & y_3 \end{array} \end{array}
$$

For each of the strategies of A, we can express the expected payoff against the strategy (y_1, y_2, y_3) of B:

$$
\begin{aligned}
A1: &\quad y_1 + 2y_2 - y_3 \\
A2: &\quad -2y_1 + y_2 + y_3 \\
A3: &\quad 2y_1 + y_3.
\end{aligned}
$$

Since player B wants to minimise the expected payoff A can achieve using any of his strategies, his objective is to keep the maximum among these three values as small as possible. This is captured by the following linear program:

$$
\begin{array}{lrcl}
\text{minimise} & v & & \\
\text{subject to} & y_1 + 2y_2 - y_3 & \leq & v \\
& -2y_1 + y_2 + y_3 & \leq & v \\
& 2y_1 + y_3 & \leq & v \\
& y_1 + y_2 + y_3 & = & 1 \\
& y_1, y_2, y_3 & \geq & 0.
\end{array}
$$

Using a linear programming solver, we obtain the optimal solution

$$
y_1 = \frac{2}{17}, \quad y_2 = \frac{8}{17}, \quad y_3 = \frac{7}{17}, \quad v = \frac{11}{17}.
$$

A similar reasoning can be used to find the optimal mixed strategy (x_1, x_2, x_3) of A. We calculate the expected payoff against every strategy of B. Since he wants to maximise the payoff, his objective is to keep the minimum of these values as high as possible:

$$\begin{array}{lrcl} \text{maximise} & u & & \\ \text{subject to} & x_1 - 2x_2 + 2x_3 & \geq & u \\ & 2x_1 + x_2 & \geq & u \\ & -x_1 + x_2 + x_3 & \geq & u \\ & x_1 + x_2 + x_3 & = & 1 \\ & x_1, x_2, x_3 & \geq & 0. \end{array}$$

Solving this LP, we obtain the optimal solution $u = \frac{11}{17}$, equal to the value of v in the previous program.

Using duality theory (see Chapter 6), one can show that the two LPs are dual to each other. The same method can be applied for arbitrary two-person zero-sum games: we obtain two LPs describing the optimal mixed strategies of the two players. These will have the same optimum value, and the two programs will be dual of each other.

13.5.1 $2 \times n$ games

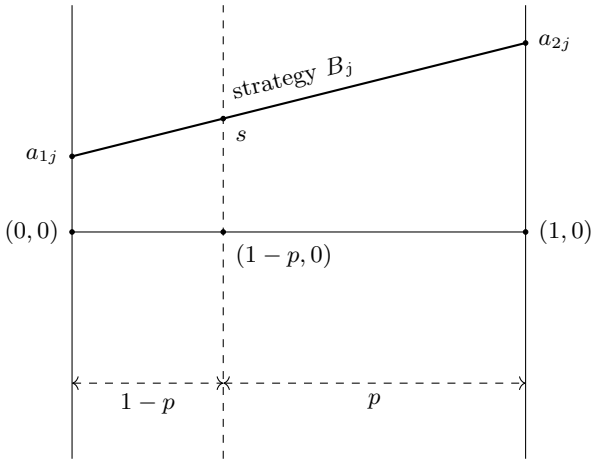
In Chapter 3, we have seen a geometric method to solve linear programs with two variables. This provides a particularly intuitive method to solve $2 \times n$ games. Consider a game, where the row player has only two strategies, while the column player has n strategies:

$$\begin{array}{cc} & \text{B} \\ & \begin{array}{cccc} 1 & 2 & \ldots & n \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \end{array} & \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \end{bmatrix} \end{array}$$

Let us denote the mixed strategy of A by $(p, 1-p)$. That is, he plays strategy A1 with probability p, and A2 with the remaining probability $(1-p)$. Then A's LP can be written as:

$$\begin{array}{lrcl} \text{maximise} & u & & \\ \text{subject to} & a_{11}p + a_{21}(1-p) & \geq & u \\ & a_{12}p + a_{22}(1-p) & \geq & u \\ & \ldots & & \\ & a_{1n}p + a_{2n}(1-p) & \geq & u \\ & 1 \geq p & \geq & 0. \end{array}$$

We use the following graphical representation. The two strategies of A correspond to two vertical lines, namely the y axis and another parallel line through the point $(1, 0)$. The j^{th} mixed strategy Bj of the column player is represented by the line connecting the points $(0, a_{1j})$ and $(1, a_{2j})$.



Consider the mixed strategy $(p, 1-p)$ of A, that is, playing A1 with probability p and A2 with probability $(1-p)$. The payoff of Bj against this strategy is $a_{1j}p + a_{2j}(1-p)$. This corresponds to the intersection of the line representing Bj, and the vertical line through the point $(1-p, 0)$, denoted by s in the picture.

We represent every strategy B_j for $j = 1, \ldots, n$ this way. Consider now a certain mixed strategy $(p, 1-p)$ of A. The *best response* strategy of B corresponds to the line that intersects the vertical line through $(1-p, 0)$ at the lowest point. (This lowest point is $(1-p, u)$ for the variable u in the LP above.) For all values of p between 0 and 1, we can identify the best responses of B with the *lower envelope* of these lines, i. e. the union of the lowermost line segments. To find the optimal mixed strategy $(p, 1-p)$ of A, we need to find the value where the best response of B is the largest possible, which corresponds to the highest point of this lower envelope. (The optimal value of the variable u.) We demonstrate this method on the following example:

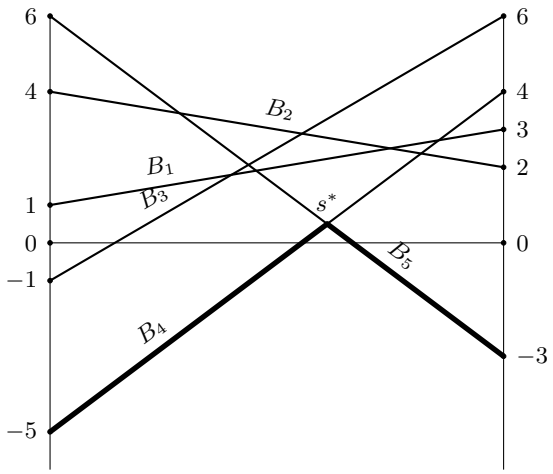
$$\begin{array}{cc} & \text{B} \\ & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \text{A} \begin{array}{c} 1 \\ 2 \end{array} & \begin{bmatrix} 1 & 4 & -1 & -5 & 6 \\ 3 & 2 & 6 & 4 & -3 \end{bmatrix} \end{array}$$

We obtain the graph on the next page. The lower envelope is denoted by the thick lines. Its highest point is denoted by s^*.

We now wish to obtain the exact optimal mixed strategies, which correspond to the intersections of the lines B_4 and B_5. That is, we have

$$\begin{aligned} -5p + 4(1-p) &= u \\ 6p - 3(1-p) &= u. \end{aligned}$$

Solving this system, we obtain $p = \frac{7}{18}$, and $u = \frac{1}{2}$. Hence the value of the game is $\frac{1}{2}$, and the optimal mixed strategy of A is $(\frac{7}{18}, \frac{11}{18})$.



To obtain B's optimal strategies, we note that these can only give positive probabilities to strategies that are *best responses* against the optimal mixed strategy of A, that is, they have the maximum expected payoff against the optimal $(\frac{7}{18}, \frac{11}{18})$ (see Exercise 13.10). Hence, B can play only B4 and B5 with positive probabilities. Associating probability q to B4 and $(1-q)$ to B5, we get

$$\begin{aligned} -5q + 6(1-q) &= v \\ 4q - 3(1-q) &= v. \end{aligned}$$

This gives $q = \frac{1}{2}$, and $v = \frac{1}{2}$. Hence the optimal mixed strategy of B is $(0, 0, 0, \frac{1}{2}, \frac{1}{2})$.

Suggested reading

Sections 14.1-5. in [HL].

13.6 Exercises

Exercise 13.1. Show that if a two-person zero-sum game has more than one saddle point, then all saddle points must have the same payoff value.

Exercise 13.2. Determine which of the games given below have pure strategy solutions and of these, which are fair. When the game has a solution in pure strategies, find optimal strategies for each player.

(a) $\begin{bmatrix} 0 & 2 \\ -1 & 4 \end{bmatrix}$ (b) $\begin{bmatrix} 5 & 0 \\ 0 & 2 \end{bmatrix}$ (c) $\begin{bmatrix} 3 & 1 \\ 4 & 0 \end{bmatrix}$ (d) $\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ (e) $\begin{bmatrix} 3 & 1 \\ -4 & 0 \end{bmatrix}$

(f) $\begin{bmatrix} 0 & 4 \\ 0 & 2 \end{bmatrix}$ (g) $\begin{bmatrix} 7 & 0 \\ 0 & 0 \end{bmatrix}$ (h) $\begin{bmatrix} 0 & 0 \\ 0 & -7 \end{bmatrix}$ (i) $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ (j) $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

Exercise 13.3. Find the optimal strategies and the value of the following game.

$$\begin{bmatrix} 3 & 5 & 4 & 9 & 6 \\ 5 & 6 & 3 & 7 & 8 \\ 8 & 7 & 9 & 8 & 7 \\ 4 & 2 & 8 & 5 & 3 \end{bmatrix}$$

Exercise 13.4. Each of two players, C and R, shows one or two fingers (simultaneously) and C pays to R a sum equal to the total number of fingers shown. Write the game matrix. Show that the game has a solution in pure strategies, and find the value and the optimal strategies.

Exercise 13.5. Each of two players, C and R, shows one or two fingers (simultaneously) and C pays to R an amount equal to the total numbers of fingers shown, while R pays to C an amount equal to the product of the numbers of the fingers shown. Construct the game matrix, and find the value and the optimal strategies.

Exercise 13.6. Two Muscovite guards (call them A and B) obtain at the canteen a sports kit. This contains a carton of Marlboros, a six-shooter, a bullet, and rules for Russian Roulette.

Each player stakes a pack of cigarettes. As senior officer present, A plays first. He may add two packs to the pot and pass the gun to B; or he may add one pack, spin the cylinder, test fire at his own head, and (God willing) hand the gun to B. If and when B gets the gun, he has the same options; he may hand it back, adding two packs of course; or he may add one pack and try the gun. The game is now ended. Each picks up half the pot if that is feasible, otherwise the bereaved picks it all up.

(a) Write the game in extensive form.

(b) The strategies of the players can be written as follows.

Player A	A1	pass;
	A2	gamble;
Player B	B1	pass, ignoring A's decision;
	B2	gamble, ignoring A's decision;
	B3	do the same what A does;
	B4	do the opposite of what A does.

What strategies should the players employ? Is the game fair?

Exercise 13.7. Find the optimal strategies and the value of the following two games. You can use any of the techniques from this chapter.

(a) $\begin{bmatrix} 2 & 2 & 1 & -2 & -3 \\ 4 & 3 & 4 & -2 & 0 \\ 5 & 1 & 2 & 5 & 6 \end{bmatrix}$ (b) $\begin{bmatrix} -1 & 4 & 3 & 3 \\ 7 & 2 & 8 & 5 \\ 1 & 5 & 3 & 6 \\ 5 & 1 & 4 & 4 \end{bmatrix}$

Exercise 13.8. Find the optimal mixed strategies for Alice and Bob in the simplified poker game.

Exercise 13.9. The firm of Gamesters Ltd. has to decide on where to build its new factory. The possible locations are towns X, Y, and Z. The profitability of their decision will be affected by the new product which their old rivals, Killjoy and Son, are due to introduce; Gamesters don't know which of products U, V, and W this will be. The possible effects on Gamester's profits are shown in the table.

		Killjoy's new product		
		U	V	W
Gamesters'	X	-3	4	2
factory	Y	7	2	1
location	Z	-2	6	3

(a) If Gamesters guess the probability of Killjoy introducing products U, V, and W as $\frac{1}{3}$, $\frac{1}{2}$, and $\frac{1}{6}$, respectively, then what should Gamesters' location decision be? Would the recommended location decision be different if a net profit of 2 or less would mean bankruptcy for Gamesters?

(b) Alternatively, suppose that Gamesters realise that Killjoy are rational opponents whose aims are the opposite of Gamesters'. Find the optimal mixed strategies for each of the following objectives:

- to maximise their profit expectation,
- to minimise their probability of bankruptcy,

Exercise 13.10. In a two-person zero-sum game, assume we are given the optimal mixed strategy $(x_1, x_2, \ldots, x_n)$ of the row player. A strategy of the column player is a *best response* to this strategy, if it has the lowest payoff against $(x_1, x_2, \ldots, x_n)$. Show that an optimal mixed strategy of the column player can only use best response strategies with positive probability.

Chapter 14

Markov chains: introduction

For the next three chapters you will need to revise the following concepts from your previous studies:

> *random variables, matrix multiplication, solving linear systems of equations by Gaussian elimination.*

Markov chains describe the behaviour of stochastic systems. The systems considered here can be in different states, and evolve over time in a probabilistic manner. The timeline is described by *discrete time steps* $t = 0, 1, 2, \ldots$. (Time may also start at $t = 1$ or other values; it makes no difference. Time may also be continuous, but that is out of the scope of this course.) A *stochastic process* is a sequence of random variables X_t, one for each time step t. For Markov chains, each variable X_t takes on values from a common *state space*. Markov chains can have state spaces that are continuous (such as the real numbers) or discrete but infinite (such as all integers), but in these lecture notes **we will assume the state space is finite**. Given that, if there are M states, we will assume the state space is $\{0, 1, 2, \ldots, M-1\}$. That will nicely fit several of our examples, like the gambler problem coming next; in other examples, like Example 14.2 with weather states *rainy*, *cloudy*, and *sunny*, we can still number the states 0 (rainy), 1 (cloudy), and 2 (sunny), so there is no loss of generality in taking the state space to be $\{0, \ldots, M-1\}$.

Example 14.1. Gambler problem. A gambler repeatedly plays a fruit machine. He has £1 at the beginning, and in every play he either wins £1 (with probability q) or loses £1 (with probability $1-q$). He finishes if either he goes broke by losing all his money, or manages to win £3, in which case he buys a pint. What are the probabilities of the two outcomes?

A stochastic process for this example can be defined for $t = 0, 1, 2, \ldots$ by

$$X_t = \text{the budget of the gambler at time } t.$$

The problem has 4 states: $0, 1, 2, 3$. At time 0, $X_0 = 1$. If either state 0 or state 3 is reached, the game is over. There is no limit to time: the game could go on forever, for example if the gambler alternates winning and losing forever (but this has probability 0). With probability 1, the game will terminate at some point, in either state 0 or 3; the question is to find the

respective probabilities. The answer will be given later, after the necessary techniques have been introduced.

An important property of the gambler's process is that the value of X_{t+1} depends only on X_t, not the prior history of the game. Indeed, for a given budget $X_t = r$ at time t, one will have $X_{t+1} = r + 1$ with probability q, and $X_{t+1} = r - 1$ with probability $1 - q$. The process is memoryless: the value of X_{t+1} depends on the value of X_t but does not depend on the values of $X_0, X_1, \ldots, X_{t-1}$.

This memorylessness is the defining property of a Markov chain:

A stochastic process is called a *Markov chain* if the next state X_{t+1} depends only the current state X_t but not on the previous history. Formally, for any time t and for any states $r_0, r_1, \ldots, r_{t+1}$,

$$\mathbb{P}(X_{t+1} = r_{t+1} \mid X_t = r_t) \;=\; \mathbb{P}(X_{t+1} = r_{t+1} \mid X_0 = r_0, X_1 = r_1, \ldots, X_t = r_t).$$

Recall the notation $\mathbb{P}(X_{t+1} = r_{t+1} \mid X_t = r)$ for conditional probability: this expresses the probability that $X_{t+1} = r_{t+1}$, provided that $X_t = r_t$. In these chapters, we will use one more restriction: these conditional probabilities do not depend on time t, just on the states:

A Markov chain is called *time-homogeneous* if, for every two states i and j,

$$\mathbb{P}(X_{t+1} = j \mid X_t = i) = \mathbb{P}(X_1 = j \mid X_0 = i), \qquad \text{for all } t = 1, 2, \ldots.$$

The gambler example is a time-homogeneous Markov chain: the probability q is the same no matter how many games have been played. In the remainder of the lectures, whenever we refer to a Markov chain, we always mean a finite, time-homogeneous Markov chain.

The transition matrix

A (finite, time-homogeneous) Markov chain can be fully described by giving the transition probability between each pair of states:

The *transition probability* from state i to state j is

$$p_{ij} = \mathbb{P}(X_{t+1} = j \mid X_t = i).$$

The transition probabilities are commonly presented in an $M \times M$ *transition matrix* P, where the jth entry in row i is p_{ij}. Both the rows and columns of P are indexed by elements of the state space $\{0, 1, \ldots, M - 1\}$.

For the gambler example, rows and column indices both run from 0 to 3 (since this is the state space), and the transition matrix is

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1-q & 0 & q & 0 \\ 0 & 1-q & 0 & q \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Note that every row in the transition matrix sums up to 1. This always holds:

$$\sum_{j=0}^{M-1} p_{ij} = 1, \quad \text{for all states } i.$$

This simply expresses the fact that from every state, we will move on to some next state (possibly the same). These are disjoint events and one of them necessarily happens, hence the sum of their probabilities is 1. In other words, the i^{th} row of the matrix P represents the probability distribution of the random variable X_{t+1} provided that $X_t = i$.

Suppose now that the initial state of the system is drawn from some probability distribution $\pi^{(0)}$, where $\pi_i^{(0)}$ denotes the probability that initially the system is at state i (that is, $X_0 = i$), $i = 0, \ldots, M-1$. What is the probability distribution $\pi^{(1)}$ for the random variable X_1? Given $j \in \{0, \ldots, M-1\}$, the probability that $X_1 = j$, provided that X_0 had been drawn from the initial distribution $\pi^{(0)}$, is given by

$$\pi_j^{(1)} = \sum_{i=0}^{M-1} \pi_i^{(0)} p_{ij}.$$

This is because, for every $i, j \in \{0, \ldots, M-1\}$, $\pi_i^{(0)} p_{ij}$ is the probability that $X_0 = i$ and $X_1 = j$. The probability that $X_1 = j$ will therefore be the sum of such probabilities over all possible values i of X_0.

If we represent $\pi^{(0)}$ and $\pi^{(1)}$ as M-dimensional row-vectors, we can therefore express $\pi^{(1)}$ in terms of $\pi^{(0)}$ by the following matrix equation:

$$\pi^{(1)} = \pi^{(0)} P.$$

Iterating the above argument, if we denote by $\pi^{(n)}$ the probability distribution of X_n provided that X_0 has been drawn at random according to the probability distribution $\pi^{(0)}$, we obtain

$$\pi^{(n)} = \pi^{(n-1)} P = \pi^{(n-1)} P \cdot P = \pi^{(n-1)} P^2 = \pi^{(n-2)} P^2 \cdot P = \pi^{(n-2)} P^3 = \cdots = \pi^{(0)} P^n,$$

where P^n denotes the n^{th} power of the matrix P.

If the initial state X_0 of a Markov chain with transition matrix P is drawn according to a probability distribution $\pi^{(0)}$, the probability distribution $\pi^{(n)}$ of X_n is given by

$$\pi^{(n)} = \pi^{(0)} P^n, \qquad n = 1, 2, 3, \ldots. \tag{14.1}$$

Let us now consider a second example.

Example 14.2. The weather in London tomorrow only depends on today's weather. There are three types of weather: rainy, cloudy, and sunny. If it is rainy today, then it will not rain tomorrow, and is equally likely to be cloudy or sunny. If it is cloudy today, then it is equally likely to be cloudy, rainy, or sunny tomorrow. If it is sunny today, then it will be rainy tomorrow. Given that it is cloudy on Tuesday, what is the probability that it will be sunny on Thursday (2 days later)?

The state transition diagram is given on the right in Figure 14.1. Let us number the states as 0 = rainy, 1 = cloudy, 2 = sunny. Then the transition matrix is

$$P = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 1 & 0 & 0 \end{pmatrix}.$$

Let us answer the question. If it was cloudy on Tuesday (putting us into row "1" of the matrix, the middle row), then on Wednesday it will be rainy, cloudy or sunny with equal probabilities $\frac{1}{3}$. In each of these three cases, respectively (all three rows of the matrix), on Thursday it is sunny (last column of the matrix) with probabilities $\frac{1}{2}$, $\frac{1}{3}$, and 0. (Note that while each row must add to 1, that is not true of the columns.) Hence, the probability of a sunny Thursday is

$$\frac{1}{3} \cdot \frac{1}{2} + \frac{1}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot 0 = \frac{5}{18}.$$

We can view this also in matrix terms. We are assuming that Tuesday's weather is known to be cloudy with probability 1. That is, the probability distribution of X_0 is $\pi^{(0)} = (0, 1, 0)$. The weather on Thursday will be given by the probability distribution $\pi^{(2)}$ of X_2. According to equation (14.1), $\pi^{(2)} = \pi^{(0)} P^2$. We can compute

$$P^2 = \begin{pmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{4}{9} & \frac{5}{18} & \frac{5}{18} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

It follows that

$$\pi^{(2)} = \pi^{(0)} P^{(2)} = (0, 1, 0) \begin{pmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{4}{9} & \frac{5}{18} & \frac{5}{18} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix} = (\frac{4}{9}, \frac{5}{18}, \frac{5}{18}).$$

Hence, if it is cloudy on Tuesday, on Thursday it will rain with probability $\pi_0^{(2)} = \frac{4}{9}$ and it will be cloudy or sunny with equal probability $\pi_1^{(2)} = \pi_2^{(2)} = \frac{5}{18}$.

Transition diagram

A good way to visualise time-homogeneous Markov chains is via the *state transition diagram*. We construct a directed graph by assigning a vertex to every state, and include an arc (i, j) if the state j can follow i with a positive probability: that is, if $p_{ij} > 0$. The graph on the left in Figure 14.1 presents this for the gambler problem and the weather problem. A Markov chain can be taught as a random walk on the graph, where from any vertex i we hop to a node j (possibly i itself) with probability p_{ij}.

14.1 The n-step transition matrix

Motivated by the weather example (Example 14.2), we now explore how to find the state probabilities multiple steps ahead.

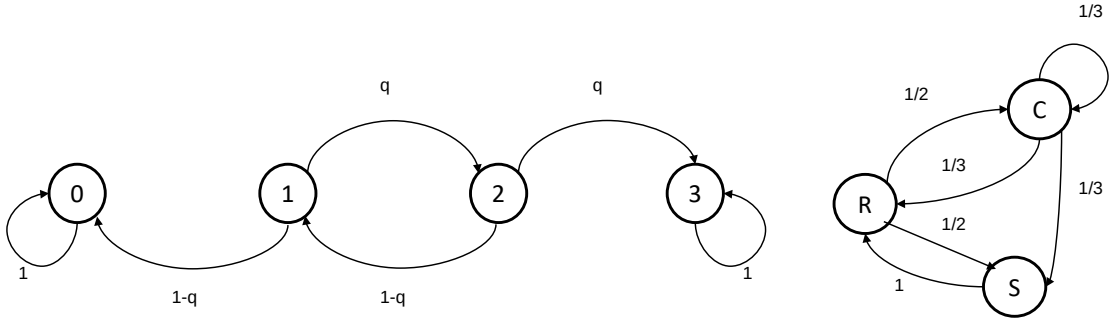


Figure 14.1: State transition diagram for the gambler problem and the weather problem.

The *n-step transition probability* from state i to state j is

$$p_{ij}^{(n)} = \mathbb{P}(X_{t+n} = j \mid X_t = i).$$

Observe that, by the memorylessness property of Markov chains,

$$p_{ij}^{(n)} = \mathbb{P}(X_n = j \mid X_0 = i).$$

Let us consider a state $i \in \{0, \ldots, M-1\}$. The i^{th} row of $P^{(n)}$ is the probability distribution $\pi^{(n)}$ of the variable X_n, provided that $X_0 = i$. That is, the i^{th} row of $P^{(n)}$ is the probability distribution of the variable X_n, provided that X_0 is drawn from the distribution $\pi^{(0)}$ where $\pi_i^{(0)} = 1$, $\pi_j^{(0)} = 0$ for every state $j \neq i$. By equation (14.1), $\pi^{(n)} = \pi^{(0)} P^n$. Observe that, for this choice of $\pi^{(0)}$, $\pi^{(0)} P^n$ is the i^{th} row of P^n. This shows that, the i^{th} rows of $P^{(n)}$ and P^n are equal for every $i \in \{0, \ldots, M-1\}$. It follows that $P^{(n)} = P^n$.

$$P^{(n)} = P \cdot P^{(n-1)} = P \cdot P \cdot P^{(n-2)} = \ldots = P^n.$$

That is, the n-step transition matrix is the nth power of the 1-step transition matrix.

Consider again the weather example. According to the above statement, we have

$$P^{(2)} = P^2 = \begin{pmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{4}{9} & \frac{5}{18} & \frac{5}{18} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

We conclude this section with a useful formula. Given to positive integer numbers $m < n$, we have $P^{(n)} = P^n = P^m P^{n-m} = P^{(m)} P^{(n-m)}$. It follows that, for any two states i and j, $p_{ij}^{(n)}$ is the scalar product of the i^{th} row of $P^{(m)}$ with the j^{th} column of $P^{(n-m)}$. Hence we have the following:

Chapman-Kolmogorov equations. For any two states i and j, and any two integers $m < n$,

$$p_{ij}^{(n)} = \sum_{k=0}^{M-1} p_{ik}^{(m)} p_{kj}^{(n-m)}.$$

14.2 Long-run properties of Markov chains

The weather example

Let us again consider the weather example, and compute $P^{(n)}$ for some larger values of n.

$$P^{(2)} = \begin{pmatrix} 0.6667 & 0.1667 & 0.1667 \\ 0.4444 & 0.2778 & 0.2778 \\ 0 & 0.5000 & 0.5000 \end{pmatrix}, \qquad P^{(5)} = \begin{pmatrix} 0.3210 & 0.3395 & 0.3395 \\ 0.3868 & 0.3066 & 0.3066 \\ 0.5185 & 0.2407 & 0.2407 \end{pmatrix},$$

$$P^{(10)} = \begin{pmatrix} 0.4104 & 0.2948 & 0.2948 \\ 0.4017 & 0.2991 & 0.2991 \\ 0.3844 & 0.3078 & 0.3078 \end{pmatrix}, \qquad P^{(20)} = \begin{pmatrix} 0.4002 & 0.2999 & 0.2999 \\ 0.4000 & 0.3000 & 0.3000 \\ 0.3997 & 0.3001 & 0.3001 \end{pmatrix}.$$

It appears that as n increases, all rows of the matrix converge to the vector $(0.4, 0.3, 0.3)$. This suggests that if we wait long enough, the weather on the first day will not matter any more: we will have 40% probability of a rainy day in the long run.

These long-run probabilities are called *steady state probabilities*. In the next chapter we shall see that under some general conditions, the probability of every state will stabilise after a large number of steps.

The gambler example

Consider now the gambler example with probability $q = \frac{1}{3}$ for winning. The long-term behaviour in this case is quite different.

$$P^{(2)} = \begin{pmatrix} 1.0000 & 0 & 0 & 0 \\ 0.6667 & 0.2222 & 0 & 0.1111 \\ 0.4444 & 0 & 0.2222 & 0.3333 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix}, \quad P^{(5)} = \begin{pmatrix} 1.0000 & 0 & 0 & 0 \\ 0.8477 & 0 & 0.0165 & 0.1358 \\ 0.5432 & 0.0329 & 0 & 0.4239 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix},$$

$$P^{(10)} = \begin{pmatrix} 1.0000 & 0 & 0 & 0 \\ 0.8567 & 0.0005 & 0 & 0.1428 \\ 0.5711 & 0 & 0.0005 & 0.4283 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix}, \quad P^{(20)} = \begin{pmatrix} 1.0000 & 0 & 0 & 0 \\ 0.8571 & 0.0000 & 0 & 0.1429 \\ 0.5714 & 0 & 0.0000 & 0.4286 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix}.$$

As n increases, the probabilities in the two middle columns will converge to 0. This is no surprise, given that the game will terminate, in the first or last state, with high probability

within the first 10 iterations. As n goes to infinity, the matrix $P^{(n)}$ will converge to

$$P^* = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{6}{7} & 0 & 0 & \frac{1}{7} \\ \frac{4}{7} & 0 & 0 & \frac{3}{7} \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

This is substantially different from the weather example, where the matrix converged to a matrix with all rows identical. The matrix P^* can be interpreted as follows: if we start the game with £1, we will eventually go broke with probability $\frac{6}{7}$ and get the pint with probability $\frac{1}{7}$. If we start with £2, the chances are better: $\frac{4}{7}$ for losing and $\frac{3}{7}$ for winning.

The following argument supports these values. If in state 1, we go to state 0 (losing immediately) with probability $\frac{2}{3}$, and we go to state 2 with probability $\frac{1}{3}$, from where we claimed losing and winning probabilities of $\frac{4}{7}$ and $\frac{3}{7}$ respectively. This is at least consistent with our claimed state-1 losing and winning probabilities. Writing out one case, the state-1 losing probability, which we claim is $\frac{6}{7}$, matches the probability of going to state 0 and thus losing immediately, namely $\frac{2}{3}$, or going to state 2 and then losing eventually, namely $\frac{1}{3} \cdot \frac{4}{7}$. Writing out the losing and winning cases for state 1 in shorter form in equations, we see consistency:

$$\text{losing: } \frac{6}{7} = \frac{2}{3} \cdot 1 + \frac{1}{3} \cdot \frac{4}{7}; \quad \text{winning: } \frac{1}{7} = \frac{2}{3} \cdot 0 + \frac{1}{3} \cdot \frac{3}{7}.$$

Similarly, from state 2 we go to state 3 and win immediately with probability $\frac{1}{3}$, and go to state 1 with probability $\frac{2}{3}$, which we claim eventually loses with probability $\frac{6}{7}$ and wins with probability $\frac{1}{7}$. This is consistent with our claimed state-2 losing and winning probabilities of $\frac{4}{7}$ and $\frac{3}{7}$ respectively:

$$\text{losing: } \frac{4}{7} = \frac{1}{3} \cdot 0 + \frac{2}{3} \cdot \frac{6}{7}; \quad \text{winning: } \frac{3}{7} = \frac{1}{3} \cdot 1 + \frac{2}{3} \cdot \frac{1}{7}.$$

We found the answer for $q = \frac{1}{3}$: the gambler can eventually reach £3 with probability $\frac{1}{7}$. However, we needed numerical experimentation to guess the right values, the answer was not fully proved, and we only got an answer for $q = \frac{1}{3}$, not all q. The general technique to answer this and similar questions will be presented in the next chapter.

Suggested reading

Sections 16.1–3 in eds. 7 and 9 of [HL].

14.3 Exercises

Exercise 14.1. A consumer who purchases one of two brands of soap powder every week is influenced by her choice of the previous week but not by earlier experience. If she purchased brand A last week, her current purchase would be of the same brand with probability 3/4 and brand B with probability 1/4. If she purchased brand B last week, the probability she would again purchase brand B is 1/2, and the probability of purchasing A is also 1/2.

Regarding the brand purchased in week t as the state of the process at time t, find the transition matrix P of the relevant Markov chain and compute P^2 and P^3.

Given that brand A is purchased in week 0, what is the probability of the following sequence of purchases from week 0 through week 4: A, B, A, A, B?

Exercise 14.2. Assume that a person's occupation may be classified as "professional", "skilled labourer", or "unskilled labourer". Assume too that of the offspring of professionals, 80% are professionals, 10% are skilled labourers, and 10% are unskilled labourers. Of the offspring of skilled labourers, 60% are skilled labourers, 20% are professionals, and 20% are unskilled labourers. Of the offspring of unskilled labourers, 50% are unskilled labourers, 25% professionals, and 25% are skilled labourers.

(a) Suppose we follow the line of descent through the eldest child, and that this line is unbroken. Set up the transition matrix for this line of descent, where time corresponds to generation and state is the occupational class. Find the probability that the grandchild of an unskilled labourer is a professional.

(b) If instead of a guaranteed unbroken line, there is a probability of 0.2 that a person will have no offspring, form a Markov chain by adding a fourth state representing the extinction of a line of descent. Find the transition matrix and again find the probability that the grandchild of an unskilled labourer is a professional.

Exercise 14.3. An electronics retailer stocks a particular tablet that can be ordered every week. The shop uses the following ordering policy. Let X_t denote the number of tablets in stock at the end of week t. If the stock is depleted ($X_t = 0$), order 3 new tablets, to arrive before the shop opens next week. Otherwise, order no new tablets.

Each week, the demand D_t follows a Poisson distribution with mean 1, and the demands in different weeks are independent random variables. Backorders are not possible: demand can be satisfied only up to stock availability. Hence the stock level at the end of week $t+1$ can be expressed as follows:

$$X_{t+1} = \begin{cases} \max\{3 - D_{t+1}, 0\}, & \text{if } X_t = 0, \\ \max\{X_t - D_{t+1}, 0\}, & \text{if } X_t \geq 1. \end{cases}$$

Note that the stock levels X_t give a Markov chain, with state space $\{0, \ldots, 3\}$.

The probabilities for the Poisson distribution are:

$$\mathbb{P}(D_t = 0) = \mathbb{P}(D_t = 1) = 0.368, \quad \mathbb{P}(D_t = 2) = 0.184, \quad \mathbb{P}(D_t \geq 3) = 0.08.$$

(a) Write the transition matrix of the Markov chain $\{X_t\}$.

(b) Draw the state transition diagram.

(c) Assume that at the end of week 0 the shop had 2 tablets in stock (that is, $X_0 = 2$). What is the probability of having exactly 1 unit in stock at the end of week 3?

Chapter 15

Markov chains: steady-state probabilities

In the previous chapter, we ended by finding long-run distributions for two Markov chain examples. Here, we will look for properties of a Markov chain that guarantee that it will have a well-defined long-run distribution, independent of the starting state; many chains of interest will have these properties. As we introduce the various properties, it may be helpful to think about why the absence of one property or another would mean that a Markov chain could fail to have such a long-run distribution.

15.1 Classes and classification of states

Classes, irreducibility, and absorbing states

Consider a time-homogeneous Markov chain with transition matrix P. We say that state j is *accessible from* state i if $p_{ij}^{(n)} > 0$ for some $n \geq 0$, that is, if we can eventually get to state j from state i with positive probability. (For convenience, we define $p_{ii}^{(0)} = 1$, so every state is accessible from itself.)

A state i from which no other state is accessible is called an *absorbing state*. Equivalently, a state i is absorbing if $p_{ii} = 1$.

In the gambler example, states 0 and 3 are both absorbing states. This expresses that once the budget runs out or reaches 3, we stay in this state forever. The process was defined to terminate when reaching one of these states; however, it is more convenient to express termination as letting the process run on, while staying forever in the same state. By the termination of a Markov chain, we mean reaching an absorbing state.

Two states i and j *communicate* if j is accessible from i, and i is accessible from j. A Markov chain is *irreducible* if every pair of states communicates.

The state transition diagram is helpful to decide which states are accessible from which others. State j is accessible from state i if and only if there is a directed path from i to j in the state transition diagram. A Markov chain is irreducible if and only if every state can be reached from every other state through a directed path, i.e., if the directed graph is "strongly connected".

In the gambler example (Fig 14.1 left), state 3 is accessible from state 2 but not vice versa; hence 2 and 3 do not communicate. Consequently, the gambler problem is *not* irreducible. On the other hand, states 1 and 2 communicate. In the weather problem (Fig 14.1 right) every state is accessible from any other, and the Markov chain is therefore irreducible.

Communication between states has the following important properties.

(i) *Reflexivity:* Each state communicates with itself.

(ii) *Symmetry:* If state i communicates with state j, then state j communicates with state i.

(iii) *Transitivity:* If state i communicates with state j, and j communicates with g, then i also communicates with g.

The first two properties directly follow from the definition. For the third property, since j is accessible from i, by definition there exists a value n such that $p_{ij}^{(n)} > 0$. Similarly, since g is accessible from j, there exists a value n' such that $p_{jg}^{(n')} > 0$. But then it follows that $p_{ig}^{(n+n')} \geq p_{ij}^{(n)} p_{jg}^{(n')} > 0$, so by definition g is accessible from i. Symmetrically, i is also accessible from g, so by definition i and g communicate.

Relations having the three properties of reflexivity, symmetry, and transitivity are known as *equivalence relations*. Such a relation enables us to partition the set of all states into disjoint classes:

> A *class* of states in a Markov chain is a maximal set of states that can mutually communicate with each other. The classes form a partition of the states: every state belongs to exactly one class.

Clearly a Markov chain is irreducible if and only if every state belongs to the same class. Classes can have just a single element. In the gambler problem there are three classes: $\{0\}, \{1, 2\}$, and $\{3\}$. The Markov chain of Figure 15.1 has three classes, $\{0\}$, $\{1, 2\}$, and $\{3, 4, 5\}$; the classes are the "strongly connected components" of the state transition diagram.

Clearly, every absorbing state forms a single-element class (as states 0 and 3 do in the gambler problem); on the other hand, not every single-element class is an absorbing state (e.g., state 0 in the previous example).

Transient and recurrent states

There is a notable difference between classes $\{1, 2\}$ and $\{3, 4, 5\}$ in the example. After the process enters the class $\{1, 2\}$, it will eventually leave: in state 2, there is a probability $1/2$ of moving to state 4, and once this happens, the process never visits states 1 or 2 again. Once the process enters the class $\{3, 4, 5\}$, it will stay inside this class forever. This motivates the following distinction between states:

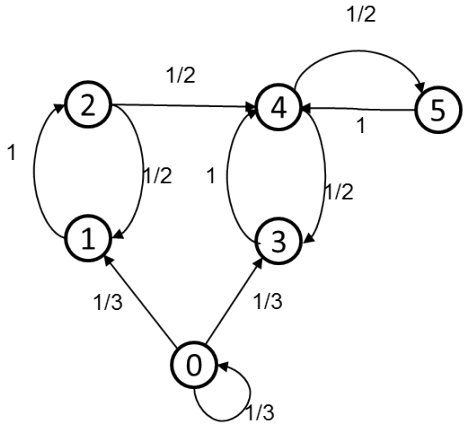


Figure 15.1: A Markov chain with three classes.

- *Transient state:* Upon entering such a state, the process might never return again. Equivalently, a state i is transient if there is a state j that is accessible from i, but where i is not accessible from j.
- *Recurrent state:* Upon entering such a state, the process will certainly return to the same state.

Hence every state is either transient or recurrent. If all states in a Markov chain are recurrent, then we say that the chain is recurrent. Furthermore, for every recurrent state in a finite Markov chain, the expected time to return is finite.[1] Any absorbing state is clearly recurrent, but not all recurrent states are absorbing. States in the same class are either all transient or all recurrent, and in particular:

Every state in a finite irreducible Markov chain is recurrent.

Note that recurrent states may belong to different classes. In the gambler problem, both states 0 and 3 are absorbing and thus recurrent, but they are in different (single-element) classes.

Periodicity

The *period* of state i is the largest integer k such that $p_{ii}^{(n)} > 0$ implies that n is an integer multiple of k. A state is called *periodic* if it has a period $k > 1$, and *aperiodic* if its period is $k = 1$. A Markov chain is *aperiodic* if all its states are aperiodic.

[1] This is not always the case in infinite Markov chains. In general, a recurrent state is said *positive recurrent* if the expected time to return is finite, and *null recurrent* if it is infinite. So for finite Markov chains all recurrent states are positive recurrent.

That is, if the system is in state i at time 0, then we can only observe state i in periods that are multiples of k. As an example, observe that, in Figure 15.1, if we start from state 3, we can only be in state 3 again in even time steps, so state 3 has a period 2. On the other hand, all states in the weather example are aperiodic. For example, if it is rainy on day 0, it cannot be rainy again on day 1, could be rainy again on day 2 (which so far looks like the previous example), but could also possibly be rainy on day 3 (for example as rain–cloud–cloud–rain) or day 4. If we can find two consecutive time periods t and $t+1$ such that the system can reach state i in both these periods, then i must be aperiodic.

15.2 Stationary and limiting distributions

Suppose that the initial state X_0 of a Markov chain is drawn at random according to some initial probability distribution π^0. This gives rise to a sequence $\pi^{(0)}, \pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \ldots$ of the probability distributions of $X_0, X_1, X_2, X_3, \ldots$. Informally, $\pi^{(0)}$ is stationary if the sequence remains constant, that is, $\pi^{(0)} = \pi^{(1)} = \pi^{(2)} = \pi^{(3)} = \cdots$. By the memorylessness of Markov chains, the sequence remains constant if and only if $\pi^{(0)} = \pi^{(1)}$. Recalling from the previous chapter that $\pi^{(1)} = \pi^{(0)}P$, we reach the following definition.

$\pi = \langle \pi_0, \ldots, \pi_{M-1} \rangle$ is a *stationary distribution* if it satisfies the *steady-state equations*

$$\begin{aligned} \pi &= \pi P, \\ \pi \mathbf{1} &= 1, \end{aligned} \tag{15.1}$$

where $\mathbf{1}$ denotes the M-dimensional column-vector of all ones.

In explicit notation, the steady-state equations can be written as

$$\begin{aligned} \sum_{i=0}^{M-1} p_{ij}\pi_i &= \pi_j \quad \text{for } j = 0, \ldots, M-1, \\ \sum_{i=0}^{M-1} \pi_i &= 1. \end{aligned}$$

Example 15.1. *Consider yet again the weather example (Example 14.2). The steady-state equations for it are*

$$\begin{aligned} \tfrac{1}{3}\pi_1 + \pi_2 &= \pi_0, \\ \tfrac{1}{2}\pi_0 + \tfrac{1}{3}\pi_1 &= \pi_1, \\ \tfrac{1}{2}\pi_0 + \tfrac{1}{3}\pi_1 &= \pi_2, \\ \pi_0 + \pi_1 + \pi_2 &= 1. \end{aligned}$$

The vector $(0.4, 0.3, 0.3)$ *is the unique solution to this system, hence it is the only stationary distribution. Note that* $(0.4, 0.3, 0.3)$ *is the vector we identified in Section 14.2*

Theorem 15.2. *Every finite Markov admits a stationary distribution. Furthermore, every irreducible Markov chain has a unique stationary distribution, and such distribution π satisfies $\pi_j > 0$ for every state j.*

Note that there are $M+1$ steady-state equations for M variables, so at least one of the equations must be redundant. The last equation cannot be redundant, since the all-zero vector satisfies the first M equations.

Another notion that we are interested in is that of limiting distribution: when is that there exists some probability distribution π such that, independent on the starting state i, the sequence $\pi^{(1)}, \pi^{(2)}, \pi^{(3)}, \ldots$ of the probability distributions of $X_1, X_2, X_3, \ldots$ converges to π?

Formally, recall that the entry $p_{ij}^{(n)}$ of the n-step transition matrix represents the probability that $X_n = j$ provided that $X_0 = i$. Hence, having a limiting distribution π means that, for every state j, independently on the starting state i, the sequence $(p_{ij}^{(n)})_n$ converges to π_j.

We say that a distribution π on the state space of a Markov chain is a *limiting distribution* if, for each state j, and for every starting state i,

$$\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j. \tag{15.2}$$

Note, in particular, recalling that $P^{(n)} = P^n$, that if a Markov chain with transition matrix P has a limiting distribution π, then,

$$\lim_{n\to\infty} P^n = \begin{pmatrix} \pi_0 & \pi_1 & \ldots & \pi_{M-1} \\ \pi_0 & \pi_1 & \ldots & \pi_{M-1} \\ & \vdots & & \end{pmatrix}. \tag{15.3}$$

Let us denote by Π the matrix on the right-hand-side of the above equation (namely, Π has M rows, all identical to the limiting distribution π). Observe that

$$\Pi = \lim_{n\to\infty} P^n = (\lim_{n\to\infty} P^{n-1})P = \Pi P,$$

which implies that the limiting distribution must satisfy the steady-state equation $\pi = \pi P$. Hence, if the limiting distribution exists, it must be a stationary distribution. Observe also that, if π' is any stationary distribution, then $\pi' = \pi' P = \pi'(\Pi P) = (\pi'\Pi)P = \pi P = \pi$, where we used the fact that $\pi'\Pi = \pi$ because all rows of Π are identical to π and because the entries of π' sum to 1. The above shows that, if a limiting distribution exists, then there are no other stationary distributions. We summarise the above discussion in the next statement.

If a Markov chain has a *limiting distribution* π, then π is the unique stationary distribution.

For example, computing the n-step transition matrices indicated that in the weather problem, each state's probability converges to a fixed value that is independent of the starting state, but that this was not true for the gambler problem.

The main reason turns out to be that the weather example is *irreducible and aperiodic*, while the gambling example is not. A fundamental theorem on Markov chains says that such a convergence holds for every irreducible aperiodic Markov chain.

Theorem 15.3 (Ergodic Theorem)**.** *Every irreducible aperiodic finite Markov chain has a* limiting distribution π*. Furthermore,* π *is the unique stationary distribution, and* $\pi_j > 0$ *for every state* j*.*

Note that the second part of the above statement is obvious from our previous discussion: if a limiting distribution exists, then it must be the unique stationary distribution; the assumption that the Markov chain is irreducible implies $\pi_j > 0$ for every state j by Theorem 15.2.

This existence and uniqueness of limiting distributions means that, if we examine the system after a large number of steps, we shall find it in state j with probability $\approx \pi_j$, no matter where we started from (and regardless of whether the number of steps happens to be even or odd, for example).

15.2.1 Necessity of conditions

Why assume that the Markov chain is irreducible and aperiodic? Let us examine these properties individually, starting with irreducibility, and see why we may fail to get the limiting probabilities guaranteed by the theorem.

If the chain is not irreducible, some state j is not accessible from some state i, and if we start in state i the probability of being in state j is 0, always, so we cannot have $\pi_j > 0$. Also, if we start from a transient state i, then by definition there is a positive probability that, starting in i, the process will never return. As time tends to infinity, the probability of observing such a state tends to 0 (sooner or later we must fail to return), again contradicting the assurance that $\pi_i > 0$.

Replacing the hypothesis that the chain is irreducible with the weaker hypothesis that it is recurrent would not suffice for having a limiting distribution. Indeed, in Exercise 15.6(b), you are asked to produce a chain that is aperiodic, recurrent, but not irreducible, and where the steady-state equations have multiple solutions (not a unique stationary distribution as desired).

Finally, consider periodicity, illustrated in the simple example of the following transition matrix

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

Every state in this irreducible Markov chain has period 3, as the state simply keeps traversing the length-3 cycle 0–1–2–0. Solving the steady-state equations yields $\pi_0 = \pi_1 = \pi_2 = \frac{1}{3}$, which makes sense: the Markov chain will "on average" be in each of the three states with equal

probability. However, the limiting-distribution requirement

$$\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j$$

does not hold. We will always have $p_{00}^{(n)} = 1$ if n is divisible by 3 and $p_{00}^{(n)} = 0$ otherwise. The chain cycles between states, rather than converging to a stationary distribution.

Suggested reading

Sections 16.4-5 in eds. 7 and 9 of [HL].

15.3 Exercises

Exercise 15.1. Consider the following one-step transition matrices of three Markov chains:

$$P = \begin{pmatrix} 0 & 0 & \frac{1}{3} & \frac{2}{3} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{pmatrix}, \quad R = \begin{pmatrix} 0 & 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & \frac{3}{4} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}.$$

For each of these Markov chains:

(a) determine the classes of states;

(b) find which states are transient, recurrent, and absorbing; and

(c) compute the period of each state.

Exercise 15.2. Prove or disprove the following claim: if all states are recurrent, then the Markov chain is irreducible.

Exercise 15.3. Show that any two states in the same class of a Markov chain have the same period.

Exercise 15.4. Demand for a certain product can be either high or low each month. In any particular month, the demand depends only on the demand in the last two months. This month's demand depends on previous demand in the following ways: If demand was high for the past two months, it will be high this month with 2/3 and low with a chance 1/3. If demand was low for the past two months, it will be high this month with probability 1/6 and low with probability 5/6. If demand last month was different from demand the month before (one high and the other low), then demand is equally likely to be high or low this month.

(a) Model this problem as a Markov chain with 4 states. Give the transition probability matrix and draw the network representation.

(b) Assume demand was low in January and high in February. What is the chance of the demand being high in April?

(c) Compute the limiting distribution.

(d) The profit from selling this product is on average £12,000 in months with high demand and £8,000 in months with low demand. What is the expected yearly profit, in the long term?

Exercise 15.5. It is currently the start of the month of May. There are four daily newspapers in the capital city: the Moon, Planet, Star, and Sun. The predicted market shares in May are 20, 30, 25, and 25% respectively. The owners of the Moon are concerned that their market share is too low, and might well lead to bankruptcy if the share dips below 15%, so they are to embark on a large advertising campaign which will come into effect immediately before the end of June. The total number of newspapers sold during June is expected to remain the same as during May, but the market shares in June are predicted to change in line with the following Markov matrix P:

From\To	Moon	Planet	Star	Sun
Moon	0.8	0.2	0.0	0.0
Planet	0.0	0.9	0.1	0.0
Star	0.1	0.0	0.8	0.1
Sun	0.0	0.1	0.2	0.7

(a) Calculate the market shares in June, assuming that the May predictions are accurate. Confirm that the Moon will be safe from bankruptcy in June.

The effect of the advertising campaign on market shares is predicted to be reflected in the following Markov matrix:

From\To	Moon	Planet	Star	Sun
Moon	0.9	0.1	0.0	0.0
Planet	0.2	0.7	0.0	0.1
Star	0.0	0.0	0.85	0.15
Sun	0.0	0.1	0.2	0.7

(b) Calculate the market shares in July and the long-term market shares, assuming that the second transition matrix continues to apply forever.

Exercise 15.6.

(a) Find all solutions to the steady-state equations in the gambler problem.

(b) Construct an aperiodic and recurrent, but non-irreducible, Markov chain, and show that the steady-state equations have multiple solutions.

Chapter 16

Markov chains: absorbing states

As in the gambler example, absorbing states can be used to represent different final outcomes of the process. If there are absorbing states the process must end in one of them, and we are interested in the probability of ending in each, which depends on the starting state. We saw in the gambler example that the n-step transition matrix converged to one with different rows, showing dependence on the starting state, while (in all rows) the columns corresponding to non-absorbing states are all 0, showing that we never end in a state other than an absorbing state. The picture is slightly more complicated when there are non-absorbing recurrent states, and we will focus on the simpler case of Markov chains with every state either transient or absorbing, as in the gambler example.

For an arbitrary state i and an absorbing state k, we define:

f_{ik} = *absorbtion probability* of being absorbed in state k, starting from state i.

Evidently, $f_{kk} = 1$ for every absorbing state k, and $f_{k'k} = 0$ if k and k' are two different absorbing states.

Assume every state is either transient or absorbing, and let K be the set of absorbing states. Then the absorbing probabilities can be obtained by solving the following system of linear equations:

$$f_{ik} = \sum_{j=0}^{M-1} f_{jk} p_{ij}, \quad \text{for all } k \in K,\ i \notin K,$$

$$f_{k'k} = \begin{cases} 1, & \text{if } k = k', \\ 0, & \text{if } k \neq k', \end{cases} \qquad \text{for all } k, k' \in K.$$

Let us compute these probabilities for the gambler's problem in the general form, with the parameter q. The two absorbing states are 0 and 3. After substituting $f_{00} = f_{33} = 1$,

$f_{03} = f_{30} = 0$, we obtain:

$$f_{10} = (1-q) + qf_{20}, \qquad f_{13} = qf_{23},$$
$$f_{20} = (1-q)f_{10}, \qquad f_{23} = (1-q)f_{13} + q.$$

One can easily compute:

$$f_{10} = \frac{1-q}{q^2-q+1}, \quad f_{13} = \frac{q^2}{q^2-q+1}, \quad f_{20} = \frac{q^2-2q+1}{q^2-q+1}, \quad f_{23} = \frac{q}{q^2-q+1}.$$

If $q = 1$ (always win), then these give $f_{13} = f_{23} = 1$; if $q = 0$ (always lose), we obtain $f_{10} = f_{20} = 1$. For the choice $q = \frac{1}{3}$, we get $f_{10} = \frac{6}{7}$, $f_{13} = \frac{1}{7}$, $f_{20} = \frac{4}{7}$, $f_{23} = \frac{3}{7}$, as in Chapter 14.

16.1 The expected number of steps

We just saw how to determine the probabilities of reaching the different absorbing states. A natural question arises: how long will it take in expectation to reach one of them - that is, how long will the process last? This can be computed similarly to the formulas above. For an arbitrary state i, let

t_i = the expected number of steps it takes to reach an absorbing state from i.

We have $t_k = 0$ for every absorbing state k.

Assume every state is either transient or absorbing, and let K be the set of absorbing states. Then the expected number of iterations to reach the different absorbing states can be obtained by solving the following system of linear equations:

$$t_i = 1 + \sum_{j=0}^{M-1} t_j p_{ij}, \qquad \text{for all } i \notin K,$$
$$t_k = 0, \qquad \text{for all } k \in K.$$

In the first equation, the "1" represents the current step, it leads to a next state j, and from j an absorbing state is reached in an expected t_j steps (with $t_j = 0$ for absorbing states j). As an illustration, assume the transition probabilities from state 1 are $p_{12} = p_{13} = \frac{1}{2}$, and it takes an expected 4 steps to reach an absorbing state from state 2, and 6 steps from state 3. Then the equation on t_1 gives:

$$t_1 = 1 + \frac{1}{2}t_2 + \frac{1}{2}t_3 = 1 + \frac{1}{2}\cdot 4 + \frac{1}{2}\cdot 6 = 6.$$

The gambler's problem with $q = \frac{1}{3}$ gives the system:

$$t_0 = t_3 = 0,$$
$$t_1 = 1 + \frac{2}{3}\cdot 0 + \frac{1}{3}t_2,$$
$$t_2 = 1 + \frac{2}{3}t_1 + \frac{1}{3}\cdot 0.$$

Solving the system gives $t_1 = \frac{12}{7}$, $t_2 = \frac{15}{7}$; hence the game will typically terminate in less than two rounds if starting with £1, and slightly more than two rounds if starting with £2.

16.1.1 The expected cost

The above computation can be easily extended to the more general setting when traversing every state i has a certain cost (or profit) c_i associated, and we are interested in computing the expected total cost (or profit) until an absorbing state is reached. We define

T_i = the expected cost until an absorbing state is reached from i.

We have $T_k = 0$ for every absorbing state k. The expected number of steps computed above corresponds to the special case when every state has equal cost $t_i = 1$. Replacing 1 in the above formulas by c_i we obtain the following:

Assume every state is either transient or absorbing, and let K be the set of absorbing states. Then the expected cost to reach the different absorbing states can be obtained by solving the following system of linear equations:

$$T_i = c_i + \sum_{j=0}^{M-1} T_j p_{ij}, \quad \text{for all } i \notin K,$$
$$T_k = 0, \quad \text{for all } k \in K.$$

16.2 The PageRank algorithm

Google's PageRank algorithm, for ranking the relative importance of different web pages, can be seen as an application of Markov chains. It is named after Google co-founder Larry Page, and was developed in the late 1990s. The basic version we study here provides an absolute ranking of web pages; for practical implementation, it needs to be further refined to give a page's importance in the context of a user's search.

We model the Internet by a directed graph, each vertex representing a web page, and an arc (i, j) denoting a hyperlink from page i to page j. As a first idea, we could say that a page is important (and should be highly ranked) if there are many hyperlinks pointing to it. But such a system could be "gamed" very easily: to promote your own web page, you could simply create numerous phantom web pages linking to it. Furthermore, a link from an important, authoritative site should count more than one from an obscure personal page. This suggests that a good ranking should have the property that:

a page is important if it is linked by important pages.

This is a circular definition but, as it turns out, it makes sense in a way well expressed by Markov chains.

To formalise the intuition on importance, assume that if a web page i has importance π_i, then it "gives" an equal share of this importance to all pages it links. That is, if web page i has ℓ_i outgoing links, each of its neighbours receives importance π_i/ℓ_i. Our model is that the importance π_j of every web page j should be equal to the importance received from all pages linking to it. That is, for all pages j,

$$\pi_j = \sum_{i \in \text{in-arcs}(j)} \pi_i \cdot \frac{1}{\ell_i}. \tag{16.1}$$

This equation is exactly the same as the steady-state equation (15.1) for a Markov chain. The Markov chain has states corresponding to web pages, and transitions (i, j) corresponding to hyperlinks from page i to page j, each such transition having probability

$$p_{ij} = \frac{1}{\ell_i}, \qquad \text{where } i \text{ has } \ell_i \text{ hyperlinks in total.}$$

Note that if π_0 is a solution, then so is $\pi = \alpha\,\pi_0$ for any real number α. Let us go ahead and impose the second steady-state equation $\pi\mathbf{1} = 1$ so as to pick a single importance measure. This "normalisation" is a natural choice as then π is a probability distribution, but if all we care about is relative ranks of web pages — comparisons between values π_i for different pages i — then it does not matter how we normalise.

The fundamental theorem of Markov chains assures us that, under reasonable conditions on the transition matrix P, the equations (16.1) and $\pi\mathbf{1} = 1$ have a unique solution π, and we take this as our definition of page rank.

The model captures the long-run browsing of a web surfer who starts from an arbitrary web page and, on each page, clicks a random hyperlink to bring up a next page. The value π_i is the long-time limit probability of being on page i.

16.2.1 Irreducibility and aperiodicity

We are done in principle, but we were glib in saying "under reasonable conditions on the transition matrix". Limiting probabilities are only guaranteed when the Markov chain is irreducible and aperiodic; let us see if the Internet graph is, and if not, what we can do.

What does it mean for the Internet graph to be irreducible and aperiodic? Let us assume for the moment that it is irreducible. Recall that Exercise (15.3) showed that all states in a class have the same period, and by definition an irreducible chain has just one class, so it is enough to show that *any* state in the Internet graph is aperiodic. That is surely true. For example, any link (A, C) along with links (A, B) and (B, C), means that if we are in state A at time t then we can be in state C at time $t + 1$ or at time $t + 2$, both with positive probability, and this implies aperiodicity of C and in turn every state.

So the real issue is whether the Internet graph is irreducible. It certainly is not, for a few simple reasons. There will be pages with no links out (called "dead ends"), pages not linked by any other, and sets of pages linked to one another but with no other links to or from the outside world (called "spider traps").

How can we patch up our random-surfer model to give an irreducible Markov chain? If we come to a dead end, let us jump to a uniformly random web page. (When the Internet

was just a novelty, and not well connected or well indexed, there were sites that would take you to a random web page; it was a way of exploring.)

To deal with pages with no in-links, and spider traps, we can use a trick called *"random teleportation"*. Fix some small constant $0 < d < 1$, and modify the random surfer model so that with probability $1 - d$ the surfer follows a random outgoing link, as in our basic model, but with probability d chooses a uniformly random web page (as in the previous paragraph).

Random teleportation gives a Markov chain that clearly is irreducible (the random jumps can get us anywhere). Since the Markov chain is also aperiodic, this guarantees a unique solution to the steady-state equations, and accordingly the ranks of the PageRank algorithm.

16.2.2 The Power Iteration Method

Computing the limiting distribution, however, is difficult: as of today there are a few billion web pages, and it is not feasible to solve the linear system by Gaussian elimination. However, the very point of a limiting distribution π was that after many steps the distribution over states converges to π *independent of the starting state or starting distribution* (see (15.2)). So to find the steady state we can simply start in any distribution $r^{(0)}$ of our choosing, and keep applying the transition matrix.

This is called the *Power Iteration Method*. Let P be the transition matrix of the Markov chain. To find π, start with an arbitrary distribution $r^{(0)}$; that is, $r^{(0)}$ is any vector, indexed by states, whose components sum up to 1 ($\mathbf{1} \cdot r^{(0)} = 1$). For example, if there are M web pages, we can take $r_i^{(0)} = 1/M$ for every i; this corresponds to starting from a uniformly random web page. Then, iteratively compute

$$r^{(t+1)} = P\, r^{(t)}.$$

The fundamental theorem guarantees that $r^{(t)}$ will converge to the stationary distribution π, as $t \to \infty$. In practice, it will converge fairly rapidly. However, even computing this matrix-vector product is challenging for matrices of such astronomical size. It requires computational cleverness, exploiting in particular the "sparsity" of P: the fact that most web pages are linked to only a few others (perhaps to tens or even hundreds, but to nowhere near all other pages).

Suggested reading

Sections 16.5–7 in eds. 7 and 9 of [HL]. The original publication of the PageRank method by Larry Page can be found on Moodle.

16.3 Exercises

Exercise 16.1. *Revisit Exercise 1.4:* What is the expected number of times you have to roll a die to get a 6? Solve the problem by using Markov chains. *(Hints: Use two states not six. What is the expected time to be absorbed in state "6"?)*

Exercise 16.2. An over-simplified model of fecundity classifies people into six states: pre-puberty, single, married, divorced, widowed, "dead or emigrated". One time period spans 7 years, and every person starts from state 0. The transition matrix is:

$$P = \begin{pmatrix} 0 & 0.9 & 0 & 0 & 0 & 0.1 \\ 0 & 0.5 & 0.4 & 0 & 0 & 0.1 \\ 0 & 0 & 0.6 & 0.2 & 0.1 & 0.1 \\ 0 & 0 & 0.4 & 0.5 & 0 & 0.1 \\ 0 & 0 & 0.4 & 0 & 0.5 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

(a) How long do people live in this model, on average?

(b) How much time do people spend in state 2 (married) on average? *(Hint: use the methodology of Section 16.1.1.)*

Exercise 16.3. *(Exam question 2015.)* An online music retailer categorises songs into six groups, corresponding to the number of downloads in the current month:

- *Group 5* Top hit: more than 1 million
- *Group 4* Very popular: 100,000–1 million
- *Group 3* Popular: 10,000–100,000
- *Group 2* Medium: 1,000–10,000
- *Group 1* Low: 100–1,000
- *Group 0* Forgotten: 0–100.

According to historical data, from month to month songs move between groups with the following probabilities:

From\To	Group 5	Group 4	Group 3	Group 2	Group 1	Group 0
Group 5	0.7	0.3	0	0	0	0
Group 4	0.3	0.5	0.2	0	0	0
Group 3	0	0.1	0.6	0.2	0.1	0
Group 2	0	0	0.25	0.5	0.25	0
Group 1	0	0	0	0.2	0.5	0.3
Group 0	0	0	0	0	0	1

For example, if a song is a top hit this month (group 5), then next month it will remain a top hit with 70% probability, or fall into the very popular category (group 4) with 30% probability. Once a song is forgotten (group 0), it remains forgotten forever.

(a) Draw the state transition diagram of this Markov chain.

(b) The song *"Break the Markov Chains!"* had 543,210 downloads this month. What is the probability that it falls into the popular category (group 3) two months from now?

(c) Consider a top hit (group 5) this month. On average, how many months does it take until this song is forgotten (first enters group 0)?

(d) The retailer pays the artists a fixed amount for every download, plus a monthly premium of £3,000 if the song is a top hit (group 5) and a monthly premium of £300 if the song is very popular (group 4). Consider a top hit this month (group 5) as in the previous part. On average, how much premium will the artist receive from the company until the song is forgotten?

Exercise 16.4. A fiendish group of LSE students tries to promote their web page www.OR-rulez.co.uk by hacking the PageRank algorithm. It is 1987 and the Internet consists of only 990 web pages. Besides their new web page, they created 9 fake pages to promote it (thus bringing the total number of web pages to 1,000). PageRank is computed using random teleportation with $d = 0.2$. Assume that the old pages do not link to the new ones, and the students cannot change this, but they can add links arbitrarily among the new pages. Pages can link to themselves. Advise them on how to add links in order to maximise the PageRank of www.OR-rulez.co.uk. What is the maximum value they can reach?

Hints: Argue that it suffices to think of a 3-state Markov chain whose states are the target new page A, the fake pages F, and the old pages Z. Think about the transition matrix without teleportation, and reason (rather than compute) what the probabilities p_{AZ} and p_{FZ} should be. Then, instead of writing out the steady-state equations in the standard form, write the equations for π_A and π_F as simply as you can (working in teleportation at this point). Find the optimal solution.

Chapter 17

Inventory models

17.1 Deterministic inventory models

A crucial decision for a shop's management is how much inventory to keep of each good and when to place replenishment orders. If a customer wants an item that is out of stock, this results in loss of revenue and also loss of goodwill. Shortage can be avoided by keeping large amounts of stock. But keeping too much stock will also be disadvantageous, due to storage costs and warehouse space requirements; further, the capital locked up in stock could have earned returns instead. Besides retailing, similar problems arise in a wide range of areas; for example, in manufacturing, production of a product may require a number of components that have to be kept in stock.

Inventory management, also called *inventory control* or *stock control*, aims to build appropriate mathematical models, and give policies for deciding the optimal level of inventory. There is a variety of models, reflecting the range of settings where inventory control problems arise. An important distinguishing feature of a model is the predictability of the demand: in *deterministic* inventory models, the exact level of demand is known ahead of time, whereas in *stochastic* inventory models, uncertainty is involved, and we assume that only the probability distributions of future demands are known. This section focuses on deterministic models, and Section 17.2 will consider stochastic models.

17.1.1 Basic concepts

Deterministic inventory models assume full knowledge of the expected demand. This can be realistic in manufacturing with a steady production rate, or for the retail of goods with consistent demands. We start with a simple example.

Example 17.1. An electronics retailer sells a certain laptop model. Demand is fairly consistent at 40 laptops per month, and the selling price to customers is £800 per laptop. Stock can be replenished from the manufacturer at £500 per laptop. The administrative cost of raising an order is £130, and the cost of transporting the order from the manufacturer's warehouse is £520, both independent of the order size. Further, keeping a laptop in stock incurs a holding cost of £6 per month, reflecting the lost returns on capital and the warehouse's insurance. The company's retail policies do not allow for shortages: all demands must be met. What is a good ordering strategy?

Let us now introduce some important notions.

- The model involves a *sequence of time periods*. In the above example, there is an indefinitely long time to plan for, with the same demand and cost every month; however, there are models with demands and costs changing over time.

- A crucial part of every model is how the *demand* is given. In the example, we have a *deterministic constant rate demand* d, meaning that the same amount d (here $d = 40$) will be sold or consumed in every period. We also assume that this demand is distributed evenly over the period.

- The *cost of ordering* an amount z can be represented as a function $C(z)$. In the simplest case this is linear: ordering every unit has a *unit cost* c, and hence the total prize is $C(z) = c\,z$. In the above example, there is also a *fixed cost* K. The cost function can be expressed as:
$$C(z) = \begin{cases} 0, & \text{if } z = 0, \\ K + c\,z, & \text{if } z > 0. \end{cases}$$
In the above example, the fixed cost is $K = 650$: this is the total administration and transportation cost; the *unit cost* is $c = 500$. More complicated cost functions are also possible, for example when ordering in bulk gives lower unit costs for larger quantities.

- The *holding cost* or *storage cost* represents all direct or indirect costs involved in keeping inventory. We will usually assume it is a linear function, assessed on a period-by-period basis. In the above example, $h = 6$ is the holding cost, that has to be paid for each unit in each period of time.

- The *shortage cost* represents the loss incurred from unsatisfied demands. A shortage may result in a lost sale or extra costs for backordering, and either way may also decrease customer satisfaction (which can also be modeled as a cost). In the above example, shortages are not allowed at all, so shortage cost is not relevant; alternatively, we could model this situation with an infinitely large shortage cost.

- When orders can be placed is described by a *review model*. Review models can be *continuous* or *periodic*. In a continuous review model, orders can be placed at any time; in a periodic review model, the stock is checked and new orders decided at fixed times, say every half a year. This was not prescribed in the example, but in this course, we will assume the more flexible continuous review model.

- The time span between when an order is placed and when the goods arrive is called the *lead time*. It was not specified in the example.

The example also specifies the selling price, £800, but, surprisingly, this is not needed in the model. If shortages are not allowed, then the selling price is irrelevant for inventory modelling because the income from selling the good will be always the same. The selling price also does not directly appear in the model even if shortages are allowed, since the loss of revenue is modeled by the shortage cost. (Instead of modeling sales as variable, we take as our baseline that all demand is met, and correct for this in the shortage cost.)

Linearity of demands and costs. The demand, holding costs, and shortage costs are given per unit (of stock) per time period. For fractional periods of time, we simply take the corresponding fraction of these quantities. In the example, if we need to store 1 laptop for 2.25 months, this will cost £13.5. We will use this simple linear model for demands and costs unless otherwise specified.

Under this assumption, with a constant demand rate and a continuous review model, the choice of time unit is arbitrary. In the example we could have chosen a year instead of a month as the time unit, giving an equivalent problem with d and h each 12 times as large (demand $d = 480$ laptops/year and holding cost $h = £72$/year per laptop). The time period only becomes relevant again in interpreting the model's output.

17.1.2 Economic Order Quantity model

The *Economic Order Quantity model (EOQ)* is the simplest inventory model, with the following assumptions:

- Demand is at a deterministic constant rate d.
- The cost of ordering $z > 0$ units is $K + cz$.
- The holding cost is h per unit per time period.
- No shortages are allowed.
- Inventory can be replenished anytime (continuous review model).
- The lead time is 0: the ordered quantity all arrives immediately after ordering.

In this model, we only need to replenish inventory when it falls to zero. Since the model is deterministic, the lead time does not make much difference. We assume the lead time is 0 for simplicity, but the same model can easily incorporate a lead time $\ell > 0$: we just place the order a time ℓ before it is needed.

We will always order a new batch of the same (optimal) size. The time between two consecutive replenishments is called a *cycle*. There are thus two parameters of interest:

- $Q =$ the *order quantity*,
- $t =$ the *cycle length*.

Of course they are not independent; given the continuous demand rate d, we get:

$$\textit{length of cycle: } t = \frac{Q}{d}.$$

Figure 17.1 shows the inventory level over time in the EOQ model.

The main goal is to determine the optimal *order quantity* Q. For a too small value of Q, the fixed ordering cost K must be paid too often, increasing the ordering cost per unit. For too large a value of Q, the holding cost can be excessive. We need to find the optimal tradeoff somewhere in between.

The cost of each cycle is the sum of its ordering and holding costs:

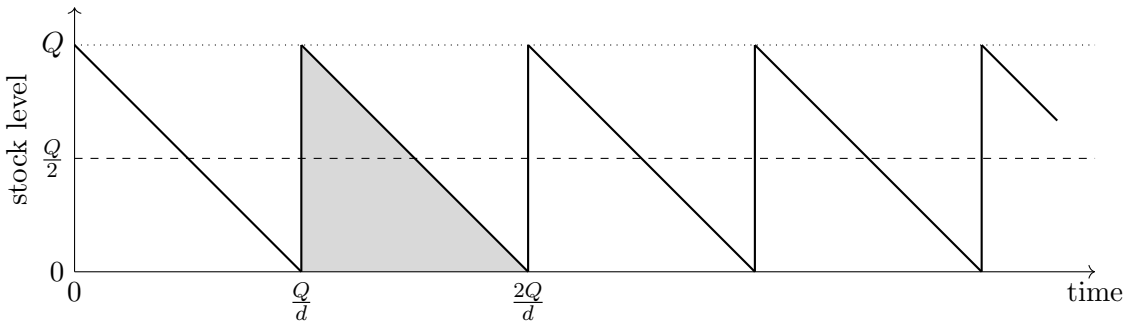


Figure 17.1: The inventory level over time in the EOQ model.

$$\begin{aligned} \textit{ordering cost per cycle} &= K + cQ, \\ \textit{holding cost per cycle} &= \frac{hQ^2}{2d}. \end{aligned}$$

To understand the holding cost, note that the average level of stock over a cycle is $\frac{Q}{2}$. The cycle length is t, so the holding cost is $h \cdot t \cdot \frac{Q}{2}$. (The value $t \cdot \frac{Q}{2}$ is precisely the integral of the stock level over the period, the area of the shaded region in Figure 17.1.)

The total cost of a cycle is the sum of these two terms: $K + cQ + \frac{hQ^2}{2d}$. We wish to minimise the (long-term average) cost per time, that is, this quantity divided by the cycle length t. (We will avoid the expression "cost per unit time" to avoid confusion with the "cost per unit" referring to stock.)

$$\textit{cost per time} = \frac{Kd}{Q} + dc + \frac{hQ}{2}$$

Our objective is thus to find the value of Q that minimises this expression. The expression is a convex function (see Figure 17.2), so the minimum occurs where the derivative is 0, i.e., where

$$-\frac{Kd}{Q^2} + \frac{h}{2} = 0.$$

This gives the optimum order quantity

$$Q^* = \sqrt{\frac{2Kd}{h}}$$

and corresponding optimum cycle length

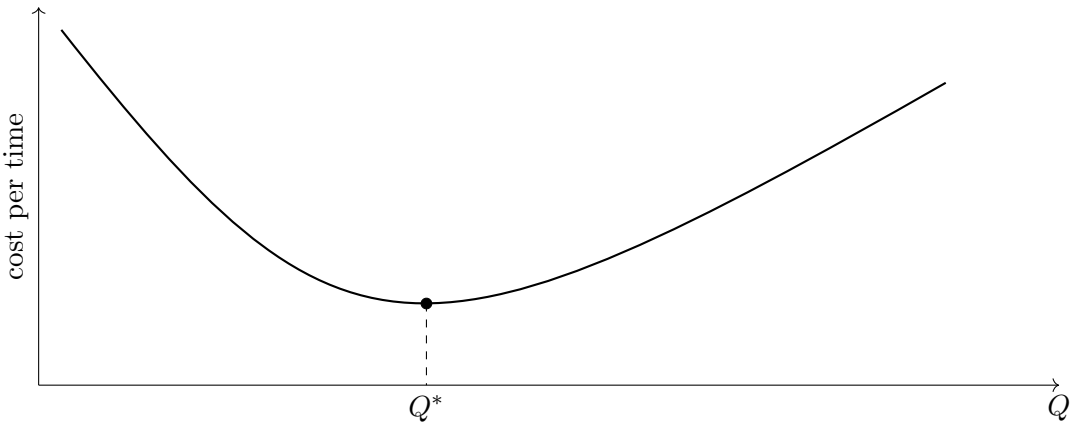


Figure 17.2: The cost per time as a function of Q.

$$t^* = \frac{Q^*}{d} = \sqrt{\frac{2K}{dh}}.$$

Observe that these values do not depend on the unit cost c. Indeed, in the cost per time expression being minimised, c appears only in the term dc, a constant independent of Q, and does not influence the choice of Q. Another way to look at it is that c is the cost per unit of stock, and (with no shortages allowed) what we must pay for stock is predetermined, and out of our control.

Let us now return to Example 17.1. The parameter values were

$$K = 650, \quad c = 500, \quad h = 6, \quad d = 40,$$

giving

$$Q^* = 93.09, \quad t^* = 2.33$$

(both values rounded to two digits) giving a cost per time of £20,558.57/month. The notionally optimal policy of ordering 93.09 laptops every 2.33 months is of course not feasible, since fractions of laptops cannot be ordered. The optimal policy we implement, then, is to order a rounded 93 laptops, adjusting the order period accordingly. Since the cost curve in Figure 17.2 has derivative 0 at Q^*, its value at some Q near Q^* is not much different. In this case, the cost per time for ordering $Q = 93$ laptops is the same to the penny as the notional optimal (larger by only about £0.0003).

17.1.3 The EOQ model with planned shortages

No shortages were allowed in the basic EOQ model. We now relax this assumption but leave the model otherwise unchanged. This models the common situation where customers may accept some delay in receiving orders. (Here the shortage is thus thought of as a backlog: an order remaining on the books, not yet filled.) We will allow a shortage cost p per unit per

time. If this is modest compared to other costs, it might be reasonable to order with shortages planned. In Section 17.2, we shall also see an inventory model with uncertain demands that can be approximated by this deterministic model.

The optimal replenishment policy will be defined by parameters Q and $t = Q/d$, as before, and one new parameter:

- S = the inventory level just after a new batch arrives and backlogged demands are satisfied.

That is, $Q-S$ units of backlogged demand are satisfied immediately upon receiving the order, leaving S units to satisfy ongoing demand. The inventory levels are as shown in Figure 17.3, a negative inventory level denoting backlogged demand.

With the order quantity Q chosen, and Q determining the cycle length t, why isn't the backlog, and therefore S, already determined? How do we control it?

As in the basic EOQ model, the ordering cost for each cycle is $K + cQ$.

Naturally, allowing shortages decreases the holding costs. Here,

$$\text{holding cost per cycle} = \frac{hS^2}{2d},$$

since the average level of inventory between the arrival of the batch and the full consumption of the stock after S/d units of time is $\frac{S}{2}$. This is an improvement: the holding cost depends on S, not Q.

However, the planned shortages mean that we now have shortage costs:

$$\text{shortage cost per cycle} = \frac{p(Q-S)^2}{2d},$$

since there will be shortage for a duration of $(Q-S)/d$ at the end of every cycle, with an average shortage of $\frac{Q-S}{2}$. The amount of stock and the amount of backlog for each cycle are the areas of the shaded triangles in Figure 17.3 above and below the horizontal axes, respectively. We get the total cost per time as the cycle cost divided by the cycle length $t = Q/d$:

$$\textit{cost per time: } C(Q,S) = \frac{Kd}{Q} + dc + \frac{hS^2}{2Q} + \frac{p(Q-S)^2}{2Q}.$$

We need to find the values of Q and S minimising this expression (with $S \le Q$). For that, we need to solve for the partial derivatives $\partial C(Q,S)/\partial Q$ and $\partial C(Q,S)/\partial S$ equal to 0. This leads to the optimum values:

$$Q^* = \sqrt{\frac{2dK}{h} \cdot \frac{p+h}{p}}, \quad S^* = \sqrt{\frac{2dK}{h} \cdot \frac{p}{p+h}}, \quad t^* = \frac{Q^*}{d} = \sqrt{\frac{2K}{dh} \cdot \frac{p+h}{p}}.$$

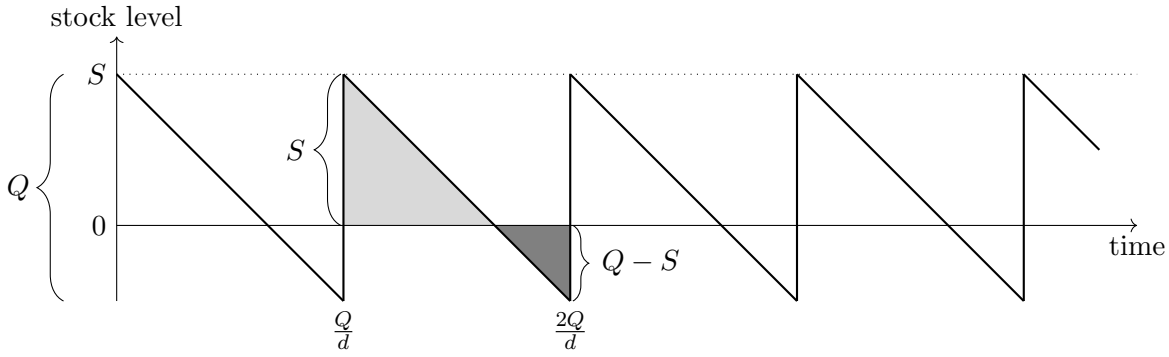


Figure 17.3: The inventory level over time in the EOQ model with shortages.

We include a derivation of the above formulas in Appendix A.1; see also Section 19.3 in [HL] 7th ed. (Sec. 18.3 in 9th ed.).

Let us now modify Example 17.1 by allowing shortages at a cost of $p = 18$ per laptop per month. In the basic EOQ model, we obtained $Q^* = 93.09$ and $t^* = 2.33$, for a cost of £20,558.57 per month. With shortages allowed, we obtain $Q^* = 107.49$, $S^* = 80.62$, and $t^* = 2.68$, for a cost of £20,483.74 per month. Rounding Q^* to 107 and S^* to 81 gives a cost of £20,483.80 per month — a negligible 6 pence per month more than the non-integral solution.

Suggested reading

Deterministic inventory models are discussed in Sections 19.1–4 in the 7th edition of [HL], and Sections 18.1–5 of the 9th edition, with examples in Sections 19.8 and 18.8 respectively.

The models above addressed the inventory problem of a single good in a single warehouse. The problems modern businesses face are much more complex. A distributor has to coordinate the flow of goods between several regional and national warehouses and field distribution centres; a manufacturer has to coordinate between a number of different production places and warehouses. Every stage in this procedure is referred to as an *echelon*, and the entire system as a *multiechelon inventory system*. An even more general problem is *supply chain management*, involving not only the inventory, but also procurement and manufacturing phases. Supply chain management can comprise a large number of echelons, and successfully-designed supply chains are instrumental for leading companies. See Section 19.8 in [HL] 7th ed. (Sec. 18.8 in 9th ed.) for the example of Hewlett-Packard.

Serial two-echelon systems are the simplest possible multiechelon system. There are two echelons, E1 and E2. The inventory at E2 has to be periodically replenished from E1. For example, E1 could be a factory and E2 a distribution centre. In a different application, E2 could be a factory and E1 another facility where the components needed for production are received. Section 18.5 in the 9th ed. of [HL] (not contained in the 7th ed.) provides the analysis of such models, already substantially more complicated than the ones described above.

17.1.4 Exercises

Exercise 17.1. An electrical mail order company, AJP Limited, sells toasters at a constant rate of 1,000 per month. AJP buys the toasters from XYZ for £10 each, and sells them for £19 each. AJP has £200,000 invested in immediate access funds at 3% p.a. (per annum). Each time an order is placed, there is a fixed delivery cost £90, and a required insurance at £10. The cost of insuring the stored goods is estimated at 2% of the purchase value p.a.

(a) How often per year should AJP order toasters from XYZ, and what should the order size be? Assume that the order lead time is negligible.

(b) How does the solution change if we also allow shortages, assuming a shortage cost 10% of the purchase value p.a.?

Hints: It might be easier to use 1 year instead of 1 month as time unit. In the holding cost, you also need account for the lost return on capital, as money is being used to purchase inventory instead of earning interest in the bank.

Exercise 17.2. Suppose the annual demand $d = 1,500$ for a product is constant over time, there is no lead-time, and the cost of placing an order is $K = 50$. The interest rate at which money can be borrowed tends to rise with the capital being borrowed, and the Finance Manager decides to model the stock-holding cost, h, as an increasing function of the average stock. Specifically, he defines it as:

$$h = a + b \cdot (\text{average stock level}),$$

where $a =$£10 per unit per year and $b =$ £0.50 per unit per year. Derive the optimality condition for the order size Q, and show that it reduces to solving a cubic equation. (You do not need to solve this equation.)

Exercise 17.3. A clothes boutique sells various versions of a dress, for which the demand is very stationary at a level of 20 dresses per day on each of the 250 days a year the shop is open. The dresses can be ordered from two small local manufacturers. Company A has higher working standards. They can make dresses at an average cost of £50 per dress, and a fixed ordering cost £300. Company B can produce dresses of a quality slightly inferior to that of Company A, so that the average cost per dress for Company B is only £40; the ordering cost is also lower, at £200. The finished dresses are sold in the boutique for £90 each, although 20% of dresses made by Company B are found to be faulty and have to be sold as seconds at less than half price: £40 each. The shop owners can purchase new orders from their investments, which exceed £1 million and currently earn them 6% p.a. Ignore all other costs and sources of income.

(a) Which company is more profitable to work with? Devise a sensible production schedule.

(b) If the selling price of the finished dresses of perfect quality is lowered to £75 each (with seconds reduced to £30 each), it is believed that the demand would increase by 40 percent. What advice would you give?

Exercise 17.4. *(Exam question 2015.)* An office supplier is selling ergonomic chairs at a price of £320. The annual demand for these chairs is fairly stable at 6,000 per year. The chairs can be ordered at a purchasing price of £200. The delivery cost of an order is £400, irrespective to its size. The order quantity is required to be an integer multiple of 20. One week elapses between the placement and the arrival of an order (1 year=52 weeks).

Warehouse costs amount to £20 per chair per year. The lost interest for capital tied up in stock is 10% of the purchase value of each chair per year. The stock is insured by an insurance company, at a rate of 5% of the purchasing price of every chair for a whole year.

(a) The company currently orders chairs ten times a year, in batches of 600. Can you advise the company to use a better ordering policy, using the EOQ model? What is the optimal order quantity size, and how many orders need to be placed per year? Determine the total annual savings of the company if they switch from the current ordering system to the one you suggested.

(b) Assuming a uniform selling rate, what is the optimal reorder point? (That is, at what level of inventory should an order be placed?)

(c) A different insurance company offers insurance for the entire inventory for a fixed annual rate of £3,000, irrespective of the stock size. Compute the total annual cost for this model, and advise the company on whether they should switch to this insurance.

(d) The supplier offers quantity discounts. The table below represents the unit price for purchasing chairs, where the price for each category applies for every unit purchased.

Quantity purchased	Price per chair
1 to 339	£200
340 to 599	£195
600 or more	£190

Does this provide the company with an incentive to change their ordering policy determined in (a)? Here we assume the original 5% insurance is used.

17.2 Stochastic inventory models

In stochastic inventory models, our knowledge and expectations about future demand are expressed by a probability distribution. You will need to revise the following concepts from your previous courses to understand this section:

random variable, probability distribution, probability density function (PDF), cumulative distribution function (CDF), expected value, normal distribution, Poisson distribution, exponential distribution.

17.2.1 Stochastic continuous review model

Our first stochastic model will be based on a simple order policy. Reordering of products will be according to the following rule for some values of R and Q:

> *Order-quantity policy:* Whenever the inventory level of the product drops to R units, place an order for Q units to replenish the inventory. The parameters to be determined are the *reorder point* R and the *order quantity* Q.

A traditional implementation of such a policy is the *two-bin system*, used for storing e.g. nuts and bolts. The items are held in two bins, the first one of size R. Units are drawn from the second bin until it runs empty; at that point a new order is placed, and until it arrives the first bin is used. Upon arrival of the order, the first bin is replenished to the level R, and the rest is put in the second bin. Today, such manual replenishment policies have been replaced by computerised inventory systems that monitor inventory levels and place orders when necessary.

The model studied in this chapter will share the following assumptions with the EOQ model with planned shortages:

- The cost of ordering $z > 0$ units is $K + cz$.
- The holding cost is h per unit per time period.
- Backorders are possible, with a shortage cost p incurred for each backordered unit.
- Inventory can be replenished at any time (continuous review model).

The following features will be different due to the stochastic nature of the problem:

- The demand is uncertain, with the *expected number* of units demanded per time unit known to be d.
- There is a lead time $\ell > 0$ between the placement and the arrival of an order.
- The demand during the lead time period is a random variable X, given by probability density function (PDF) $f(x)$.

Here we will often use continuous random variables to represent the demand, although in reality the demand should be integer. However, continuous variables typically give a good approximation for integer demands, and can be easier to deal with. Given the above information, the goal is to devise an optimal order-quantity policy, that is, determine the optimal reorder point R and order quantity Q.

Choosing the order quantity Q

There is no general formula to express the optimal choice of Q. To obviate this issue, we imply consider the EOQ model with planned shortages, and a constant demand rate d. The formula from the previous section gives

$$Q = \sqrt{\frac{2dK}{h} \cdot \frac{p+h}{p}}.$$

While the above choice is not the best one in general, Axsäter [Ax] shows that the expected cost is at most 12% higher with the above choice, if the reorder point R is chosen optimally for the given Q. While in principle such optimal R can be computed, this falls outside the scope of these lectures.

Choosing the reorder point R

Whereas the lead time was not important in the deterministic models, it plays a key role in stochastic models. For lead time 0, we could set $R = 0$, as the new order immediately arrives when the stock runs out. We now have to take into account the demand distribution over the lead time. We have not yet established the criteria to judge how good a certain choice of R can be. Several different measures could be used; the one we choose here is the following:

> *Service level requirement:* the probability of a stockout between the time an order is placed and received should not be more than q. The value q will be called the *tolerated stockout probability.*

To keep the holding cost low, we shall aim for the smallest possible value of R under this condition. Let us start with an example.

Example 17.2. The lead time for orders in a bicycle shop is 10 days for a certain road bike. According to historical data, the demand during these 10 days is uniform between 21 and 40. That is, all values 21, 22, ..., 39, 40 occur with equal probability $\frac{1}{20}$, and no other values are possible. (Such a distribution is not very realistic, but it is just for the sake of illustration.) The company desires that the probability of a stockout should be no more than 15%.

We have $q = 0.15$, and should set a level R such that the probability $\mathbb{P}(X > R) \leq 0.15$, where X is the number of bicycles arriving. This gives $R = 37$; indeed, $\mathbb{P}(X > 37) = \frac{3}{20} = 0.15$, whereas for $R = 36$ the probability of stockout would be 20%.

Another important quantity is the expected amount of stock at the time the new batch arrives. Note that the stock value may be 0 in case of stockout; however, a certain amount may be left in case of lower demand.

> The *safety stock* is the expected inventory level just before the new stock arrives: $R - \mathbb{E}(X)$, where X is the demand during the lead time.

In the above example, $\mathbb{E}(X) = 30.5$, and therefore the safety stock is $37 - 30.5 = 6.5$.

In general, we can compute the reorder point R determined by the service level q, and the corresponding safety stock, using the probability density function $f(x)$ of the lead-time demand X:

For a given service level requirement q, choose the reorder point R such that

$$\mathbb{P}(X > R) = q,$$

which can be written as

$$\int_R^\infty f(x)dx = q.$$

Let us recall the definition of a cumulative distribution function (CDF), namely $F(d) = \mathbb{P}(X \leq d)$. For a continuous distribution, the CDF is given by

$$F(d) = \int_0^d f(x)dx.$$

In the present context, $F(d)$ denotes the probability that the lead-time demand is at most d. By the nature of complementary events, $\mathbb{P}(X > d) = 1 - F(d)$, so the service-level condition above can be rewritten as:

$$F(d) = 1 - q.$$

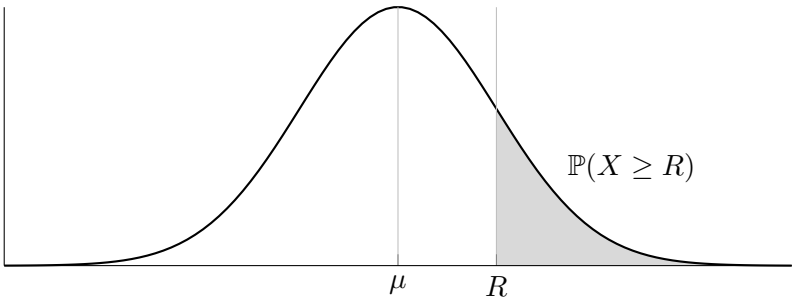


Figure 17.4: A normal distribution with the tail probability for R.

See Figure 17.4 as an illustration of this value for a normal distribution $N(\mu, \sigma^2)$.[1] You do not have to compute these integrals in most cases; for common distributions, CDF values are given in tables in statistics handbooks, and can be computed with standard software such as Excel. For example, assume X has a normal distribution with mean $\mu = 30$ and standard deviation $\sigma = 3$, and the requirement is $q = 0.05$. From the tables for the standard normal distribution $N(0, 1)$, we get that the value r for which $F(r) \approx 0.95$ is $r = 1.65$. Then we can compute

$$R = \mu + r\sigma = 30 + 3 \cdot 1.65 = 34.95.$$

Since R needs to be integer, we shall set the reorder point at 35.

[1]The demand will be integer, whereas a normal random variable is almost always fractional. Nevertheless, the normal is often a good approximate model for discrete quantities with continuous distributions.

Let us mention that instead of the service level requirement, various other measures can also be used, for example an upper bound on the average delay in filling backorders in case of stockout. This is not discussed here, but can be implemented in a similar manner.

Example

Let us revisit Example 17.1 from the previous section, allowing a shortage cost $p = 2$. The demand was assumed to be deterministic at $d = 40$; we now assume it follows a Poisson distribution at a monthly rate of 40. We will also assume that the lead time is 0.5 months, and the tolerated stockout probability is $q = 10\%$.

The defining property of the Poisson distribution is that events occur at a given *rate*, randomly, and this applies in the lead time of 0.5 months just as it does over a whole month. With a monthly rate of 40, the half-month lead time has rate $\lambda = \frac{1}{2} \cdot 40 = 20$, so the demand X in this time is Poisson with parameter $\lambda = 20$, and therefore $E(X) = \mathrm{Var}(X) = 20$.

We already computed the optimal order quantity $Q^* = 107$ using the EOQ model with planned shortages in the previous lecture. We need to find the value of R with $\mathbb{P}(X > R) \leq 0.1$. This can be easily computed by Excel: the function

$$\text{POISSON.DIST}(R,20,\text{TRUE})$$

returns the cumulative probability distribution $\mathbb{P}(X \leq R)$. The smallest value of R with cumulative distribution > 0.9 will be $R = 26$. Consequently, the optimal reorder policy will be to order 107 new laptops once the stock falls below 26.

17.2.2 The newsvendor problem

The previous models addressed inventory problems over a longer period of time. We will now look at a different kind of inventory model, suitable for problems involving *perishable goods*. Planning is for a single time period only, but units left unsold at the end expire or lose value. The name derives from the standard example: a newspaper is a perishable product as it becomes outdated in a day. The newsvendor needs to order an appropriate amount of stock so that it can meet all or most demand; at the same time, only a small amount should be left unsold. The problem would be trivial for a deterministic demand, but more realistically the demand is uncertain, and will be modelled by a random variable.

Besides newspapers, perishable products can include flowers, produce, restaurant food, fashion goods, or phones and laptops when a new model is introduced. Another important example is airplane tickets: demand shortfall means wasted empty seats; excess demand means overbooking and costly "bumping" of ticketed customers. Hotel booking is a similar case, and the technique developed for such problems is *Revenue management*, including flexible pricing (also mentioned in Chapter 1).

Here, we will focus on the simplest case of inventory management for perishable goods. Our model is based on the following assumptions:

- There is a single good and a single time period involved.
- The good must be sold in this time period, and excess amounts lose all or part of their value.

- The demand is uncertain, given by a random variable X with probability density function $f(x)$.
- The unit purchase cost is c.
- For every unit of unmet demand, there is a shortage cost p, representing lost revenue and loss of customer goodwill.
- For every unsold unit, there is a holding cost h, representing storage costs less salvage value (salvage being the amount that can be recovered from selling a stale item). Note that h can be negative if the salvage value exceeds the storage cost.
- There is no initial inventory. (*This is without loss of generality. Why?*)

The only decision to be made is:

> Optimise the amount S of the good to be purchased.

As before, we do not take the selling price explicitly into account, but sales revenue is implicitly accounted for. The baseline condition is that we precisely satisfy the demand (and the selling price figures into the profit we make). The shortage costs and holding costs are the differences with respect to this baseline.

The *unit cost of underordering* is $p - c$: for each unit we could have sold, if only we had bought it, we suffer the shortage cost p, but we save the purchase price c. (Note that $p > c$: p includes the lost revenue, i.e., the sale price, which exceeds the purchase price.) Likewise, *the unit cost of overordering* is $h + c$: for each unit we bought but cannot sell, we suffer the holding cost as well as the wasted purchase cost. (Note that while h may be negative, $h + c > 0$: otherwise our "loss" from underordering is negative, meaning it is more profitable to sell a perished good than a fresh one.)

If we order S units, the amount sold is given by the smaller of X and S:

$$\min\{X, S\} = \begin{cases} X & \text{if } X < S, \\ S & \text{if } X \geq S. \end{cases}$$

Hence the total cost is given by the random variable:

$$C(X, S) = cS + p \max\{0, X - S\} + h \max\{0, S - X\}.$$

We now let $C(S) = \mathbb{E}[C(X, S)]$ be the expected cost resulting from ordering S units. In terms of the probability density function $f(x)$, we have:

$$\begin{aligned} C(S) = \mathbb{E}[C(X, S)] &= \int_0^\infty C(x, S) f(x)\, dx \\ &= \int_0^\infty \big(cS + p \max\{0, x - S\} + h \max\{0, S - x\}\big) f(x)\, dx \\ &= cS + p \int_S^\infty (x - S) f(x)\, dx + h \int_0^S (S - x) f(x)\, dx. \end{aligned} \tag{17.1}$$

This is again a convex function (like that in Figure 17.2), and we can find the minimum by setting its derivative to 0:

$$c - p\int_S^\infty f(x)\,dx + h\int_0^S f(x)\,dx = 0.$$

Computing the derivative is not immediate but not hard; think about it, or see Section 19.6 of the 7th edition of [HL], Appendix 3 of the 9th edition. Just as in the Section 17.2.1, we rewrite this in terms of the CDF:

$$c - p(1 - F(S)) + hF(S) = 0.$$

Therefore the optimal value S^* satisfies

$$F(S^*) = \frac{p-c}{p+h}. \tag{17.2}$$

Since $F(S)$ is a continuous function increasing monotonically from 0 to 1, there will be a unique value S^* satisfying this equation.

$F(S^*)$ is called the *optimal service level*: if it is, e.g., 0.7, the demand will be met in 70% of the cases. Recall that $p - c$ is the unit cost of underordering, call it c_u and $h + c$ is the unit cost of overordering, call it c_o, so we can also write the optimal service level as

$$F(X^*) = \frac{c_u}{c_o + c_u}.$$

With demand continuously distributed we are bound either to overorder or underorder, and it is interesting (and easy to check from the equation above) that the optimal service level balances the probability of overordering times the unit cost for doing so, with the probability of underordering times the unit cost for that: $F(S^*)\,c_o = (1 - F(S^*))\,c_u$. Exercise 17.7 hints at how we could have reasoned this way to start with, giving an alternative derivation of (17.2).

Example 17.3. A haulage company has a fleet of 35 lorries to transport goods around the country. It costs £200 per day to run a lorry, whether it is needed or not. (This might include insurance, depreciation, road fund licence, and maintenance.) It is always possible to hire other lorries from suppliers for £350 per day each. The (stochastic) demand per day is for X lorries, with X exponentially-distributed with mean 45. The company is planning to downsize its fleet. How many lorries would you advise them to sell?

Setting up the model is the trickiest part. Imagine starting from a clean slate: how many lorries should you "buy" for the day? Each lorry costs $c = 200$. Each excess lorry has "holding cost" $h = 0$: there is no extra cost associated with not using it, nor is there any salvage value (we're not able to rent it out to someone else). The unit cost of "overordering" (overcapacity) is just $c_o = h + c = c = 200$. Each lorry short of our needs has shortage cost $p = 350$, the cost of renting from a supplier. The unit cost of "underordering" is $c_u = p - c = 150$, the

extra amount this rental costs above what you could have paid. The optimal service level is $c_u/(c_o + c_u) = 150/(150 + 200)$.

Recall that the exponential distribution with mean 45 has PDF

$$f(x) = \tfrac{1}{45} e^{-x/45}$$

on the interval $[0, \infty)$. The CDF is

$$F(d) = \int_0^d \tfrac{1}{45} e^{-x/45} dx = 1 - e^{-d/45}.$$

Hence we need to find S^* such that

$$1 - e^{-S^*/45} = F(S^*) = \frac{150}{350} = \frac{3}{7}.$$

Reordering and taking the logarithm we get

$$-\frac{S^*}{45} = \ln\left(\frac{4}{7}\right),$$

therefore

$$S^* = 25.18.$$

Hence for the given demand, the optimal number of lorries to maintain is 25; the company is advised to sell 10 of their lorries to maximise profit.

Initial inventory and fixed costs

The model above assumed that the ordering cost is linear, at unit cost c. We now extend it to the more general case where we consider fixed costs: ordering $z > 0$ units will cost $K + cz$. As a further extension of the model, we also assume that some *initial inventory* $I \geq 0$ is already present. The newsvendor therefore has to choose between two options:

1. Do not order at all, but only use the initial inventory.
2. Order some quantity $T > 0$ at price $K + cT$.

Let S denote the total stock after the order arrives. That is, $S = I$ if no order is placed, and $S = I + T$ if an order is placed. Let $C(S)$ denote the cost associated with a stock level S as in (17.1) for the basic newsvendor problem seen previously. That formula included a fixed cost cS for ordering S units. Since we receive the initial I units for free, we have to subtract cI for our model. Therefore the total cost $\overline{C}$ can be obtained in the two cases as

$$\overline{C}(S) = \begin{cases} C(I) - cI, & \text{if no order is made: } T = 0 \text{ and } S = I; \\ C(S) - cI + K, & \text{if a quantity } T = S - I > 0 \text{ is ordered.} \end{cases}$$

If we choose not to order, there is nothing further to choose: S is I. If we choose to order, the value of S minimising $\overline{C}$ is the same as the value of S minimising C, namely S^*, since $\overline{C}$ and C differ by a constant ($-cI + K$ is independent of S). If $S^* \leq I$, we choose not to order: we already have more than the optimal inventory. Otherwise, the question is which cost is

smaller: $\overline{C}(I) = C(I) - cI$ (from not ordering), or $\overline{C}(S^*) = C(S^*) - cI + K$ (from ordering). Since both contain the term cI, the answer is that if $C(I) \leq C(S^*) + K$ then we are better off not ordering, while if $C(I) > C(S^*) + K$ then we should order an amount $S^* - I$ to bring inventory up to the optimal level.

To view it another way, if $S^* > I$, then — ignoring the fixed cost for a moment — by bringing inventory up to level S^* we reduce cost by $C(I) - C(S)$. But we do so at the expense of the fixed cost K, so it is worthwhile only if $C(I) - C(S) > K$, which is equivalent to the condition just given.

To summarise, the optimal order policy is the following:

- If $S^* \leq I$ then *do not order*;
- if $C(I) \leq C(S^*) + K$ then *do not order*;
- if $S^* > I$ and $C(I) > C(S^*) + K$ then *order* $S^* - I$ *units.*

Suggested reading

Stochastic inventory models are discussed in Sections 19.5–6 in the 7th edition of [HL], and in Sections 18.6–7 of the 9th edition.

17.2.3 Exercises

Exercise 17.5. XYZ sell a wide range of products. The daily demand for one such product has a normal distribution $N(20, 36)$, i.e., with mean 20 and variance 36. If stock falls to a fixed level R, an order is placed with the local wholesaler for a fixed quantity Q, which will arrive exactly five days later. The cost of placing an order is £45. The wholesaler charges £30 per unit but XYZ can usually sell it for £49.99. Stockholding costs average out at 8% per unit per year. If demand cannot be met from stock then it is held over until the order arrives, and XYZ gives the customer a discount of £1 per unit. Management has determined 8% as the tolerated probability of stockout before the order arrives.

XYZ are open for business 50 weeks in the year, Monday to Saturday. What are the values of R and Q which will minimise XYZ's expected annual variable stock-related costs and respect the tolerated stockout probability for this product?

Exercise 17.6. A London-based discount warehouse, CheepDeels Ltd., is open to the public 365 days a year and sells television sets manufactured by the company Sonny. CheepDeels buys the sets at £200 each and sells them for £250 (whereas the selling price quoted in Sonny's own internet site is £290 each). Each time an order for stock is placed, it costs £100 for insurance, £60 for customs paperwork, and £840 to transport the sets from Sonny's storage site in Portsmouth. The whole process takes a month on average, and this time may be considered to be constant. The total stockholding cost of keeping one set in stock for one month is estimated to be £20.

The monthly demand for sales of these television sets at CheepDeels is approximately as follows:

Number of sets:	50	55	60	65	70	75	80	85
Frequency:	5	10	30	28	15	8	2	2

At what values should CheepDeels fix the reorder level R, and the order quantity Q, if their company policy is to run out of stock at most once a year on average?

Exercise 17.7. Jack is trying to raise money by selling flowers and plants to motorists waiting at the lights at Holborn station. Among other things, he sells bunches of orchids for £10, each bunch containing eight orchids that he has bought from the market earlier that day for 50p each. When the rush hour is over, if he still has some bunches of orchids left, he can always sell them for £2 to other vendors outside the local hospital.

Over the last 60 days the numbers of bunches of orchids that his customers (actual or potential) have requested have been as follows:

Number of bunches:	6	7	8	9	10	11	12	13	14
Number of days:	3	8	9	12	10	7	5	4	2

Estimate the expected demand for his bunches of orchids per day. Ignoring any other costs or revenues, deduce that he should purchase enough orchids each day to make up either 10 or 11 bunches, and decide whether or not the eleventh bunch is likely to improve his profit.

Exercise 17.8. A university publisher revises a certain textbook every four years. It has been three years since the textbook was revised. There are currently no copies left in stock. The publisher must determine how many copies of the book should be printed for the next year (2016). The sales department believes that sales during 2016 are governed by the distribution in the next table.

Copies demanded	3,000	4,000	5,000	6,000
Probability	0.3	0.2	0.4	0.1

Each copy of the book sold during the next year brings the publisher £35 in revenue. Any copies left at the end of 2016 cannot be sold at full price but can be sold at £5 online. The cost of a printing is £50,000 plus £15 per book printed.

(a) How many copies of the textbook should be printed?

(b) Would the answer change if we assume 2,000 copies are already in stock from last year?

(c) Assume that the revision of the textbook is postponed by a year, to 2018; the old edition will be used for one more year, in 2017. Hence the copies remaining at the end of 2016 can be sold in 2017 at full price, with £35 revenue per copy. Copies left at the end of 2017 can be sold at £5. The estimated sales will follow the same distribution in 2017 as in 2016, given in the table above; the demand in 2017 will be independent from that in 2016. The publisher decides to print all copies for the two years now. The cost of printing is as above, and there is currently no stock. How many copies of the textbook should be printed?

Chapter 18

Queueing theory

18.1 Introduction

Queueing (also spelled "queuing") theory is the study of queues. These are situations in which units (possibly customers) arrive for a service and must wait if the service facility is occupied. When the service rate is too low or the arrival rate is too high, excess waiting results; if the service capacity is high relative to the arrival rate, idle capacity results. In order to balance the costs associated with undercapacity and overcapacity, it is necessary to understand the behaviour of the queueing system. We will be interested in questions such as the expected number of customers in the system, the expected time that a customer will spend in the system, or the fraction of time that a server will be idle (i.e., not serving). Queueing theory is widely applicable, and queues and queueing models can take many forms.

A simple queueing system is sketched in Figure 18.1. A queue will have one or more servers, customers who arrive in some known pattern, and a service-time distribution. The servers will usually form parallel channels, some or all of which may be specialised (for example, "cash payment only" or "fewer than ten items" in a supermarket). Queues may also be linked in series, such as when a customer must pass through several servers before completing service, or in a manufacturing assembly line. Different queues may interfere with one other, as at traffic lights.

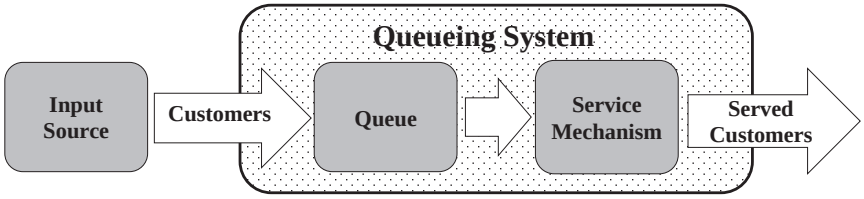


Figure 18.1: A simple queueing system schematic diagram.

There will usually be a known interarrival time distribution for the "customers", but the arrival intervals may or may not be independent. Where one queue feeds into another, the

interdeparture pattern from the first is of interest, because it is the interarrival pattern for the second. Arrivals may also occur in batches, to enter a theatre for example. Service times may also be dependent, such as when the server is a machine which has to be set up for each job, and the setup time depends on both the last job completed and the job which is about to be started. In the real world there may be a system for reducing service times in emergencies or if the queue gets too long, and there may be a priority rule applying in the queue.

The customer population may be finite or infinite, and may be split into different categories, for example with separate airport border-control queues for "U.K. or European Community citizens" and "other nationals". Customers may queue as expected, but may also balk (not join the queue), renege (leave the queue), jockey (look for a shorter queue), or collude (have one person hold a spot in line for his partners).

Queueing systems typically have some initial, transient period in which the starting conditions are relevant, but then exhibit a steady-state behaviour independent of the initial condition (much like Markov chains). Queueing theory usually focuses on the steady state.

Queueing theory is not an optimisation method. It also cannot analyse all possible queueing situations. Where applicable, though, queueing theory gives exact, analytical solutions to queueing systems; even when not directly applicable, it can give general insight into queueing systems. For systems beyond the range of queueing theory, *simulation* techniques can be used to estimate properties of the system. Simulation has become widely used as growing computing power has made this more feasible and software has improved.

Suggested reading for Queueing Theory is Chapter 17 of [HL] (editions 7 and 9), Sections 1–6 (through the M/M/1 queue part of the Birth-and-Death section).

Appendices to these Lecture Notes A number of appendices relevant to this topic are included at the end of these lecture notes. Appendix A.2 discusses the Poisson process and how it relates to the Poisson and exponential distributions. Appendices A.3 and A.4 give expressions for finite and infinite sums of geometric or arithmetic-geometric progressions, which will be used several times.

18.2 Key factors influencing the life cycles of queues

Three key factors influence the life cycles of individual queues:

- The arrival patterns of the "customers", or "items".
- The logic of the queue behaviour.
- The characteristics of the service facility.

18.2.1 The Queueing System

- **A queueing system** (in these notes also referred to simply as a *system*) is comprised of the people queueing to be served, and the people being served.

  For example, suppose there are currently ten people in a shop: nine customers and the shopkeeper. The shopkeeper is serving one customer; two others are queueing, waiting to be served; and six are wandering around the shop, choosing goods to buy. In this case the queueing system consists of only three people: the customer who is being served

and the two who are queueing. The system does not include the shopkeeper (the server, or servers) nor the customers who are still shopping.

- **State of the system** = number of customers in the queueing system. (To repeat, this is the number of customers queueing plus the number being served.)

18.2.2 Input source and arrival process

The input source of the queueing system is the set of potential customers who *may* enter the system. We will assume that the **input source has infinite size**. This is both for mathematical convenience (calculations are far easier in the infinite case) and because the assumption is often realistic, namely in the case where the population is very large.

We need to specify the probability distribution according to which customers join the queue. This is specified by the probability distribution of the *interarrival times*, that is, the time duration between two consecutive arrivals. We denote by $a(t)$ the probability density function of the interarrival times.

A common assumption, which will be made throughout these notes, is that the arrival process is a *Poisson process*. It is conventional to denote by λ the Poisson *arrival rate*, that is, the expected number of arrivals (or generally "events") per unit time (perhaps an hour, or a day). It is hard to overstate the importance of the Poisson process, and it has several equivalent definitions. We will use a definition based on the following two assumptions:

(a) Arrivals in disjoint (non-overlapping) time intervals are independent; this is called the *independent increment property*. In particular, this implies *memorylessness*: the arrival process after time t is independent of the arrivals before time t.

(b) For $\delta > 0$ "small" and for every time t, the number of arrivals in any time interval $[t, t+\delta)$ is 1 with probability essentially $\delta\lambda$ (i.e., this to within an additive term of order smaller than δ), 0 with probability essentially $1 - \delta\lambda$, and more than 1 with probability essentially 0.

Assume that the arrival process follows assumptions (a) and (b), and that λ is the arrival rate. Let $X(t)$ be the random variable denoting the number of arrivals in the time interval $[0, t)$.
Then the **number of arrivals** $X(t)$ in $[0, t)$ follows the *Poisson Distribution* with parameter λt, that is,

$$\mathbb{P}(X(t) = n) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}.$$

The mean number of arrivals in the time interval $[0, t)$ is

$$\mathbb{E}[X(t)] = \lambda t.$$

The **interarrival times** follow the exponential distribution with rate λ, mean $1/\lambda$ (both parametrisations are common), with PDF

$$a(t) = \lambda e^{-\lambda t}.$$

A proof of these statements is given in Appendix A.2.

For example, if between 9am and midnight your mobile phone receives messages randomly at a typical rate of 3 messages per hour, this would be modeled by a Poisson process with rate $\lambda = 3$ (or, in units, 3/hour), with expected interarrival time $1/\lambda = 1/3$, i.e., 1/3rd an hour, or 20 minutes.

The Poisson process models cases where arrivals are independent of previous arrivals, and at most one customer arrives at any instant. This is often a good model (especially with the flexibility of time-dependent Poisson processes that allow the rate λ to vary over time), but not always. In practice there can be bulk arrivals, such as groups of people coming to a restaurant. Arrivals can depend on previous ones through the number of people already in the queue, for example with people balking if the queue is too long. And arrivals can depend on the time since the last arrival, for example with trains running on a regular schedule rather than coming randomly.

18.2.3 Queue Discipline

Queue discipline determines the order in which customers are served. Here we mention a few common examples:

- **FIFO**: First-in first-out, also known as "first-come first-served".
- **LIFO**: Last-in first-out. This is also thought of as a "stack": the last item put on top of the stack (into the queue) is the first to be taken off.
- **Random**: The next customer in queue to be served is selected at random. This happens, for example, when passengers are put on hold at the airport.
- **Priority queueing**: Customers are classified at arrival into different classes, with different priority levels. Within each category, customers are served FIFO. This is routinely done in hospital A&E departments.

The simplest queue discipline is FIFO, but the discipline does not affect many quantities of interest, including the distribution of the queue length, the throughput, or the server idle time, since for all of these, customers are all equivalent. The discipline does affect the waiting-time distribution (think of FIFO, where waiting times should be relatively equal, and LIFO, where you might get served almost immediately from the top of the stack, or might have a terribly long wait if you wind up buried in the stack).

18.2.4 Service Characteristics

We denote by μ the *service rate*. We will assume that service times follow an exponential distribution with parameter μ, thus having PDF

$$s(t) = \mu e^{-\mu t}.$$

An advantage of using this exponential model is the following property of the exponential distribution.

> The minimum of k independent exponentially-distributed random variables with parameters $\mu_1, \ldots, \mu_k$ is an exponential random variable with parameter $\mu_1 + \mu_2 + \cdots + \mu_k$.

This property is very useful in analysing queueing systems with multiple servers, for example when there are multiple independent servers all having exponential service-time distribution with the same parameter μ. In this case, if n people are being served simultaneously, the service time of the system is the minimum of the service times among the n busy servers, and the service rate is the sum of the service rates, that is, $n\mu$.

While mathematically convenient, the assumption of exponentially-distributed service times is often unrealistic, for example if the service time is roughly the same for every customer.

18.2.5 Queue Development

A queueing system is described by the number of people in the system at any given time.

Let us denote by $p_n(t)$ the probability that there are n customers in the system at time t, for $n = 0, 1, 2, 3, \ldots$. Typically, there is a *transient phase* (e.g., just after a shop opens, before many customers have entered) where the distributions $p_n(t)$ vary considerably with time t, but (as in Markov chains) this leads on to a *steady state* where limit probabilities

$$\lim_{t \to +\infty} p_n(t) = p_n$$

exist for all n. The probability distribution p_n, $n = 0, 1, 2, 3, \ldots$ is the *steady state distribution.*

In the most general case, arrival and service rates depend on the number of people in the system. We will use the following notation.

λ_n = mean arrival rate (expected number of arrivals per unit time) at state n, $n = 0, 1, 2, 3, \ldots$.

μ_n = mean service rate (expected number of services completed per unit time) at state n, $n = 1, 2, 3, \ldots$.

Note that μ_0 is not defined, because no customer is served if the system is empty. Common scenarios are ones where λ_n is increasing in n (e.g., customers attracted to a successful restaurant) or decreasing in n (e.g., customers balking if they see long queues).

18.3 Little's law

Before we proceed, we present Little's law. Named for John D.C. Little, who proved a version of it in 1961, the law holds in great generality for arrival-departure processes. We consider any system where customers arrive over time, starting at time 0, spend some time in the system, and then leave the system. We denote by $N(t)$ the number of arrivals in the system in the time interval $[0, t]$, and by $L(t)$ the number of customers in the system at time t. We denote by W_n the amount of time that the nth customer to arrive spends in the system. Note that here we make no assumption at all on the probability distributions of the interarrival times nor on the time spent in the system. Little's law gives a general relationship between the following three quantities:

- the arrival rate, $\lambda = \lim_{t \to +\infty} \dfrac{N(t)}{t}$, i.e., the average number of arrivals over time;

- the average time spent in the system, $W = \lim_{n \to +\infty} \frac{1}{n} \sum_{j=1}^{n} W_j$;
- the average number in the system, $L = \lim_{t \to +\infty} \frac{1}{t} \int_0^t L(\tau)\, d\tau$.

Little's law: Assuming that λ and W exist and are finite, $L = \lambda W$.

The law is uninteresting if there is only a finite number of arrivals, for then $\lambda = 0$ and $L = 0$. The law also says nothing if the arrival rate exceeds the service rate, since then the servers cannot keep up with the customers joining the system, the queue grows infinitely, and both L and W are infinite.

18.4 The birth-and-death process

In these notes we will assume that the inputs and outputs of the queueing model behave according to a *birth-and-death process*, where a birth corresponds to an arrival in the system, and a death corresponds to a served customer. As shown in Figure 18.2, transitions are only from state n to $n+1$ (birth), and from state n to $n-1$ (death).

Assumptions of birth-and-death process:

1. In state n, $n = 0, 1, 2, \ldots$, interarrival times (of births) are exponentially-distributed with rate λ_n (mean $1/\lambda_n$).
2. In state n, $n = 1, 2, \ldots$, service times (leading to deaths) are exponentially-distributed with rate μ_n (mean $1/\mu_n$).
3. The interarrival times and service times are all mutually independent.

We will rely on the *memoryless* property of the exponential distribution: conditioned upon the event not having happened by some time $\tau > 0$, the distribution of the remaining time until the event occurs follows the same exponential distribution. (For example, if light bulb lifetimes were exponentially-distributed with mean 1 year, then the expected remaining lifetime of an 8 months old bulb would still be 1 year.) In notation, if X is an exponentially-distributed random variable with rate λ, then $(X - \tau \mid X > \tau)$ is also exponentially-distributed, with the same rate. Since a rate-λ exponential has PDF $\lambda e^{-\lambda t}$, its probability density near $t = 0$ is λ. That is, for a small value dt, given that the event has not occurred by time t, the probability it occurs in the interval $(t, t + dt)$ is essentially $\lambda\, dt$. We will use this property in formulating the balance equations in the next section.

18.4.1 The balance equations

Recall that, in steady state, the probability $p_n(t)$ of being in state n at time t is a constant p_n independent of the time t. This means the following stationarity condition.

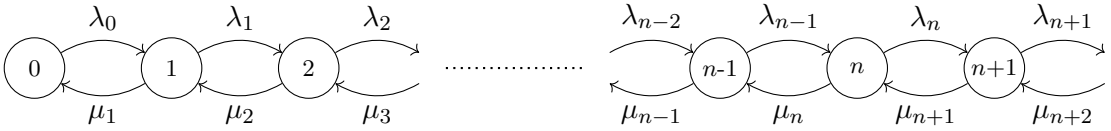


Figure 18.2: Representation of the birth-death process.

"Rate in = rate out" principle. In the steady state, for any state $n = 0, 1, 2, \ldots$ of the system, in an infinitesimal time period $(t, t + dt)$, the probability of entering state n is equal to the probability of leaving state n.

As we shall see, this principle will give rise to *balance equations* which fully describe the steady state probabilities p_n, $n = 0, 1, 2, \ldots$.

By the consequence of memorylessness we just established, if there are n customers in the system at time t, then in the time interval $(t, t + dt)$:

(i) the probability that a customer leaves the system is $\mu_n dt$;
(ii) the probability that a customer enters the system is $\lambda_n dt$.

Let us first derive the balance equations for the state $n = 0$. In the time interval $(t, t+dt)$:

- by (i), the probability that the system was in state 1 and goes to state 0 is $p_1\mu_1 dt$;
- by (ii), the probability that the system was in state 0 and goes to state 1 is $p_0\lambda_0 dt$.

Therefore, by the "rate in = rate out" principle, in the steady state we have $p_1\mu_1 dt = p_0\lambda_0 dt$. Dividing by dt throughout, we obtain the balance equation for state 0:

$$p_1\mu_1 = p_0\lambda_0.$$

Similarly, consider any other state $n \geq 1$. As shown in Figure 18.2, there are two transitions entering n and two leaving it. Writing $a \to b$ for the event that we were in state a and transition to b in the time interval $(t, t + dt)$, we have that:

$$\begin{aligned}
\mathbb{P}(n - 1 \to n) &= p_{n-1}\lambda_{n-1}dt \\
\mathbb{P}(n + 1 \to n) &= p_{n+1}\mu_{n+1}dt \\
\mathbb{P}(n \to n + 1) &= p_n\lambda_n dt \\
\mathbb{P}(n \to n - 1) &= p_n\mu_n dt.
\end{aligned}$$

Applying the "rate in = rate out" principle and dividing through by dt gives the balance equation:

$$p_{n-1}\lambda_{n-1} + p_{n+1}\mu_{n+1} = p_n\lambda_n + p_n\mu_n.$$

Table 18.1 shows the collection of all the balance equations. How can we solve for the stationary state probabilities p_n? It can be done straightforwardly, but there is an observation that makes it easier and more intuitive.

State	Balance equation
$n = 0$	$p_1\mu_1 = p_0\lambda_0$
$n = 1$	$p_0\lambda_0 + p_2\mu_2 = p_1\lambda_1 + p_1\mu_1$
$n = 2$	$p_1\lambda_1 + p_3\mu_3 = p_2\lambda_2 + p_2\mu_2$
$\vdots$	$\vdots$
n	$p_{n-1}\lambda_{n-1} + p_{n+1}\mu_{n+1} = p_n\lambda_n + p_n\mu_n$
$\vdots$	$\vdots$

Table 18.1: Balance equations for the steady state of the birth-death process.

The rates into and out of state 0 must balance, but these are the same as two of the rates into and out of state 1 (see Figure 18.2). Therefore, of the four rates into and out of state 1 that must balance, the pair to and from state 0 must balance, and therefore the pair to and from state 2 must also balance. Likewise, of the four rates to and from state 2 that must balance, we just showed that the pair between 2 and 1 must balance, thus the pair between 2 and 3 must also balance. We conclude that there is not just balanced flow into and out of each state, but balanced flow between each *pair* of states, a property often called *detailed balance*.

The same thing can also be seen from Table 18.1. Flipping the $n = 0$ equation and subtracting it from the $n = 1$ equation gives the detailed balance equation

$$p_2\mu_2 = p_1\lambda_1.$$

Flipping that and subtracting it from the $n = 2$ equation gives

$$p_3\mu_3 = p_2\lambda_2,$$

and so forth.

The $n = 0$ balance equation gives $p_1 = \frac{\lambda_0}{\mu_1}p_0$, the $n = 1$ detailed balance equation gives $p_2 = \frac{\lambda_1}{\mu_2}p_1 = \frac{\lambda_0\lambda_1}{\mu_1\mu_2}p_0$, and the $n = 2$ detailed balance equation gives $p_3 = \frac{\lambda_2}{\mu_3}p_2 = \frac{\lambda_0\lambda_1\lambda_2}{\mu_1\mu_2\mu_3}p_0$. It is clear that we have the following general pattern.

In the steady state, for $n = 1, 2, 3, \ldots$, the probability that n customers are in the system is

$$p_n = C_n p_0, \tag{18.1}$$

where we define

$$C_n := \frac{\lambda_0\lambda_1\lambda_2\cdots\lambda_{n-1}}{\mu_1\mu_2\mu_3\cdots\mu_n}. \tag{18.2}$$

Since $p_0 + p_1 + p_2 + p_3 + \cdots + p_n + \cdots = 1$, it follows that

$$p_0\left(1 + C_1 + C_2 + C_3 + \cdots\right) = 1,$$

therefore we have shown the following.

In the steady state, the probability that no customers are in the system is

$$p_0 = (1 + C_1 + C_2 + C_3 + \cdots)^{-1}. \tag{18.3}$$

Once the values of the arrival and departure rates λ_n and μ_n are known, assuming the steady state exists, p_0 can in principle be calculated using (18.3), whereupon the values of p_n for $n = 1, 2, 3, \ldots$ are determined by (18.1).

Many queueing systems are based on the birth-and-death process. This means that their input and the service mechanism satisfy assumptions 1, 2, 3 of the birth-and-death process. We will consider several cases, depending on whether there is only one server or multiple servers and on whether or not the arrival rates depend on the number of customers in the system (the state).

18.5 Arrival rates independent of the state

In this section we assume that the arrival rates λ_n are independent of the state n of the system. This means that

$$\lambda_n \equiv \lambda, \quad n = 0, 1, 2, 3, \ldots$$

This assumption implies that the queueing system has infinite capacity: if capacity were capped, then the arrival rate would have to drop to 0 if the system was filled to capacity, as no more customers are allowed in.

We further assume that all servers are "identical", in the sense that each server has the same rate μ. Taking as an example two servers, when there is no customer in the system there is of course no service; when there is one customer, only one server can be at work (this is an assumption) and therefore service progresses at rate μ; when there are two or more customers, both servers can act and service progresses at rate 2μ.

18.5.1 Single server

We first consider the case of a single server. In this case, the service rates μ_n are also independent of n (for $n > 1$), and we take

$$\mu_n \equiv \mu, \quad n = 1, 2, 3, \ldots$$

It will be convenient to define the *traffic intensity* to be

$$\rho := \frac{\lambda}{\mu}.$$

The traffic intensity ρ is the ratio of the mean number of arrivals and mean number of customers served per unit time. Note that we need to assume that $\rho < 1$, otherwise the arrival rate is greater than the service rate and the system will never reach a steady state because the queue will increase indefinitely.

In this case, the coefficients C_n defined in (18.2) in the previous section become

$$C_n = \left(\frac{\lambda}{\mu}\right)^n = \rho^n.$$

(Note that the troublesome μ_0 does not appear in any C.) Then, (18.3) gives

$$p_0 = (1 + \rho + \rho^2 + \rho^3 + \cdots)^{-1} = 1 - \rho.$$

The last equality follows from the well known geometric series formula $\sum_{n=0}^{\infty} \rho^n = 1/(1-\rho)$; see equation (A.4) in Appendix A.3. We conclude the following.

The steady-state probability that there are no customers in the system is

$$p_0 = 1 - \rho. \tag{18.4}$$

By equation (18.1), we obtain the following.

The steady-state probability that n customers are in the system, for $n = 1, 2, 3, \ldots$, is

$$p_n = \rho^n(1 - \rho). \tag{18.5}$$

We are now set up to derive several **key indicators** of the queue in this single-server setting.

1. The **probability that the server is idle** is $p_0 = 1 - \rho = 1 - \frac{\lambda}{\mu}$.
2. Thus, the **proportion of time the server is busy** is $1 - p_0 = \rho$.
3. Let S denote the **expected number of customers being served**. If X is a random variable representing the number of customers being served at any given moment (so X equals either 0 or 1), then

   $$S = \mathbb{E}[X] = 0 \cdot p_0 + 1 \cdot (1 - p_0) = \rho.$$

4. Let L_S denote the **expected number of customers in the system**. Then, using (18.5),

   $$L_S = 0p_0 + 1p_1 + 2p_2 + \cdots = \sum_{n=0}^{\infty} np_n = \sum_{n=0}^{\infty} n(1-\rho)\rho^n = (1-\rho)\sum_{n=0}^{\infty} n\rho^n = \frac{\rho}{1-\rho}.$$

   The last equality follows from the well-known arithmetic-geometric series formula $\sum_{n=0}^{\infty} n\rho^n = \rho/(1-\rho)^2$; see equation (A.5) in Appendix A.4.

   Alternatively, substituting $\rho = \lambda/\mu$, we can express L_S as

   $$L_S = \frac{\lambda}{\mu - \lambda}.$$

5. Let W_S denote the **expected time in the system**. By Little's law,

$$W_S = \frac{L_S}{\lambda} = \frac{1}{\mu - \lambda}.$$

6. Let L_q denote the **expected number of customers in the queue**. (Remember that this does not include the one customer, if any, being served.) Then

$$L_q = L_S - S = \frac{\rho}{1-\rho} - \rho = \frac{\rho^2}{1-\rho}.$$

7. Let W_q denote the **expected queueing time**. Consider the queue itself as a system; departing from the queue means starting to get service. Applying Little's law to the queue,

$$W_q = \frac{L_q}{\lambda} = \frac{\rho^2}{\lambda(1-\rho)} = \frac{\lambda}{\mu(\mu-\lambda)},$$

where the last equation follows from substituting $\rho = \lambda/\mu$.

An alternative derivation is from the expression for W_S previously obtained, using that $W_S = \mathbb{E}[\text{queueing time} + \text{service time}] = W_q + \frac{1}{\mu}$. This gives

$$W_q = W_S - \frac{1}{\mu} = \frac{1}{\mu - \lambda} - \frac{1}{\mu} = \frac{\lambda}{\mu(\mu-\lambda)}.$$

8. Let w_q denote the **conditional expected queueing time**, the expected time a customer has to queue given that there is already a queue when s/he enters the system. Remember that p_0 is the probability that there is no queue. By the law of total expectation, the (unconditoned) expected queue length is

$$\begin{aligned} W_q &= \mathbb{P}(\text{no queue})\mathbb{E}[\text{queue length} \mid \text{no queue}] + \mathbb{P}(\text{queue})\mathbb{E}[\text{queue length} \mid \text{queue}] \\ &= p_0 \cdot 0 + (1-p_0) \cdot w_q. \end{aligned}$$

Hence

$$w_q = \frac{W_q}{1-p_0} = \frac{1}{\mu - \lambda},$$

where the second equality follows from $W_q = \frac{\lambda}{\mu(\mu-\lambda)}$ and $p_0 = 1 - \frac{\lambda}{\mu}$.

Example 18.1. Fred Smith runs a small newsagents shop. At around midday, when the local school has its lunch hour, his only customers are school children. All the goods are on or behind the counter, so a child is either queueing or being served. On average a child enters the shop every 40 seconds during this period and can be served in 30 seconds.

The arrival rate is $\lambda = 90$ customers per hour, and the service rate is $\mu = 120$ customers per hour, so the traffic density is $\rho = 0.75$.

1. The probability that the Fred is idle is $p_0 = 1 - \rho = 0.25$.

2. The proportion of time that Fred is busy is 0.75.

3. The expected number of children being served is $\rho = 0.75$.

4. The expected number of children in the shop is $L_S = \frac{0.75}{1-0.75} = 3$.

5. A typical child's expected time spent in the shop is $W_S = \frac{L_s}{\lambda} = \frac{3}{90} = \frac{1}{30}$ hours, that is, 2 minutes. (Alternatively, $W_S = \frac{1}{120-90} = \frac{1}{30}$.)

6. The expected queue length is $L_q = 3 - 0.75 = 2.25$.

7. A typical child's expected time spent in the queue is $W_q = \frac{L_q}{\lambda} = \frac{2{,}25}{90} = \frac{1}{40}$ hours, that is, 90 seconds.

8. A typical child's expected time spent queueing if someone is already in the shop when s/he arrives is $w_q = \frac{1}{\mu-\lambda} = \frac{1}{30}$ hours, that is, 2 minutes.

18.5.2 Multiple servers

Note that, when we have multiple servers, even though each has the same service rate μ, the service rate μ_n of the whole system depends on the number n of customers in the system. Indeed, when $n \leq s$ only n servers are busy, so the service rate is $n\mu$, whereas when $n > s$ all s servers are busy, so the service rate is $s\mu$. That is,

$$\mu_n = \begin{cases} n\mu, & n = 1, \ldots, s \\ s\mu, & n = s+1, s+2, \ldots \end{cases}$$

To compute the steady state distribution p_n, $n = 0, 1, 2, \ldots$, we need to compute the coefficients C_n appearing in equation (18.1).

For $n \leq s$,

$$C_n = \frac{\lambda_0\lambda_1\lambda_2\cdots\lambda_{n-1}}{\mu_1\mu_2\mu_3\cdots\mu_n} = \frac{\lambda^n}{(1\mu)(2\mu)\cdots(n\mu)} = \frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n.$$

For $n > s$, we have

$$C_n = \frac{\lambda_0\lambda_1\lambda_2\cdots\lambda_{n-1}}{\mu_1\mu_2\mu_3\cdots\mu_n} = \frac{\lambda^n}{(1\mu)(2\mu)\cdots(s\mu)\underbrace{(s\mu)\cdots(s\mu)}_{n-s \text{ times}}} = \frac{1}{s!}\left(\frac{1}{s^{n-s}}\right)\left(\frac{\lambda}{\mu}\right)^n.$$

We define the *traffic intensity* to be

$$\rho = \frac{\lambda}{s\mu},$$

so that the system is stable if $\rho < 1$, and the queue increases indefinitely if $\rho \geq 1$. Then we can write C_n in terms of ρ, obtaining the following.

In the steady state, for $n = 1, 2, 3, \ldots$, the probability that n customers are in the system is

$$p_n = C_n p_0,$$

where

$$C_n = \begin{cases} \dfrac{(s\rho)^n}{n!}, & n = 1, \ldots, s \\ \dfrac{(s\rho)^s}{s!}\rho^{n-s}, & n = s+1, s+2, \ldots. \end{cases} \tag{18.6}$$

From equation (18.3) we can calculate the value of p_0. In order to do this, we need to compute

$$\sum_{n=1}^{\infty} C_n = \sum_{n=1}^{s} \frac{(s\rho)^n}{n!} + \frac{(s\rho)^s}{s!} \sum_{n=s+1}^{\infty} \rho^{n-s}.$$

Little can be done to simplify the first sum, but it is simply the finite sum of s terms. For the second sum, we have

$$\sum_{n=s+1}^{\infty} \rho^{n-s} = \sum_{i=1}^{\infty} \rho^i = \rho \sum_{i=0}^{\infty} \rho^i = \frac{\rho}{1-\rho}.$$

Therefore, we have shown the following.

The steady state probability that there are no customers in the system is

$$p_0 = \left(1 + \sum_{n=1}^{s} \frac{(s\rho)^n}{n!} + \frac{(s\rho)^s}{s!} \frac{\rho}{1-\rho} \right)^{-1}. \tag{18.7}$$

We now get the **key indicators** of the queue in this multiple-server setting.

1. **Expected proportion of idle time per server.** When there are $n < s$ customers in the system, only n of the s servers will be busy, so the average proportion of idle servers is $1 - \frac{n}{s}$. Thus the expected proportion of idle time per server is

$$\sum_{n=0}^{s-1} \left(1 - \frac{n}{s}\right) p_n.$$

   This is simply a sum of s terms, hence it can be computed explicitly for any given problem.

2. L_q = **expected number in the queue.** If $n \leq s$, then the queue is empty, whereas if $n > s$ then there are $n - s$ people in the queue. Therefore, with p_0 given by (18.7),

$$\begin{aligned} L_q &= 1p_{s+1} + 2p_{s+2} + 3p_{s+3} + \cdots \\ &= p_0(C_{s+1} + 2C_{s+2} + 3C_{s+3} + \cdots) \\ &= p_0 \frac{(s\rho)^s}{s!}(\rho + 2\rho^2 + 3\rho^3 + \cdots), \end{aligned}$$

and applying the arithmetic-geometric sum formula gives

$$L_q = p_0 \frac{(s\rho)^s}{s!} \frac{\rho}{(1-\rho)^2}, \tag{18.8}$$

3. W_q = **expected waiting time in the queue.** By Little's law,

$$W_q = \frac{L_q}{\lambda},$$

where L_q is given by (18.8).

4. W_S = **expected time in the system**. As before, this is

$$W_S = \mathbb{E}[\text{queueing time + service time}] = W_q + \frac{1}{\mu} = \frac{L_q}{\lambda} + \frac{1}{\mu}.$$

5. L_S = **expected number of customers in the system.** By Little's law, this is

$$L_S = \lambda W_S = L_q + \frac{\lambda}{\mu}.$$

6. S = **expected number of customers being served.** This is the expected number of people in the queueing system minus the expected number of people in the queue, that is

$$S = L_S - L_q = \frac{\lambda}{\mu}.$$

Example 18.2. Fred Smith has hired an assistant to help him in his shop when children are waiting to buy. As before, Fred can serve a customer in 30 seconds, and his new assistant is equally fast on average. As before, children enter the shop every 40 seconds, on average.

In this example, the number of servers is $s = 2$, the arrival rate is $\lambda = 90$ customers per hour, as before, and service rates are $\mu_1 = 120$ customers per hour if there is only one customer in the system, and $\mu_n = 240$ customers an hour if there are $n \geq 2$ customers in the system. The traffic density is $\rho = \frac{90}{240} = \frac{3}{8} = 0.375$.

According to (18.7),

$$p_0 = \left(1 + \left[0.75 + \frac{(0.75)^2}{2!}\right] + \left[\frac{(0.75)^2}{2} \cdot \frac{0.375}{1 - 0.375}\right]\right)^{-1} = (2.2)^{-1} = \frac{5}{11}.$$

1. The expected proportion of idle time per server is

$$p_0 + (1 - \frac{1}{2})p_1 = p_0 + \frac{1}{2}p_1 = p_0(1.375) = 0.625.$$

2. The expected number in the queue is

$$L_q = \frac{5}{11}\frac{(0.75)^2}{2}\frac{0.375}{(1-0.375)^2} \approx 0.123$$

   people.

3. The expected waiting time is

$$W_q = \frac{L_q}{90} = 0.0013$$

   hours, that is, about 4.9 seconds.

4. The expected time in the system is $W_S = W_q + \frac{1}{\mu} = 4.9 + 30 = 34.9$ seconds.

5. The expected number of customers in the system is $0.123 + 0.75 \approx 0.872$.

6. The expected number of customers being served is $S = 90/120 = 0.75$.

18.6 Arrival and/or service rates dependent on the state

In the following, we give examples of systems where the arrival and service rates depend on the number of customers in the system. In all these examples, we will assume that the queueing system has finite capacity.

18.6.1 Single server

Example 18.3. Example 18.1 is extended as follows (here we assume that Fred is the only server). Fred Smith is becoming concerned about the volume of shoplifting going on in his shop, and decides to limit the number queueing or being served to 3. Because the shop is more attractive to these larcenous children when Fred is busy serving a child, the arrival rate increases to 120 per hour when someone is already in the shop, and stays at 90 when the shop is empty. For what fraction of his time is Fred now idle? How many customers will he expect to lose because of this strategy? How long is the queue, on average?

Here there is a "finite waiting room" situation because at most 3 children are allowed in the system (and no others are allowed to be in the shop browsing). The value of n, the number in the system, is thus one of 0, 1, 2 or 3 only.

We have that $\lambda_0 = 90$, whereas $\lambda_n = 120$ children/hour, for $n = 1, 2$, but $\lambda_n = 0$ for $n \geq 3$. For every state n, the service rate is $\mu = 120$ as in Example 18.1.

We must calculate p_0, p_1, p_2 and p_3; we know that $p_n = 0$ for $n \geq 4$. By Equation 18.1, we have that:

$$\begin{aligned} p_1 &= \frac{\lambda_0}{\mu_1} p_0 = \frac{90}{120} p_0 = \frac{3}{4} p_0; \\ p_2 &= \frac{\lambda_0 \lambda_1}{\mu_1 \mu_2} p_0 = \frac{90 \cdot 120}{120 \cdot 120} p_0 = \frac{3}{4} p_0; \\ p_3 &= \frac{\lambda_0 \lambda_1 \lambda_2}{\mu_1 \mu_2 \mu_3} p_0 = \frac{90 \cdot 120 \cdot 120}{120 \cdot 120 \cdot 120} = \frac{3}{4} p_0. \end{aligned}$$

Since we know that the sum of the probabilities must be 1, we have

$$p_0(1 + \frac{3}{4} + \frac{3}{4} + \frac{3}{4}) = 1,$$

which gives

$$p_0 = \frac{1}{1 + \frac{9}{4}} = \frac{4}{13}, \quad p_1 = p_2 = p_3 = \frac{3}{4} \cdot \frac{4}{13} = \frac{3}{13}.$$

Note that the computation of p_0 has not required us to sum an infinite series. This is always the case when there is a finite waiting room.

Fred is now idle for just over 30 percent of his time ($p_0 = \frac{4}{13} \approx 0.31$), whereas previously he was idle for only 25 percent of his time. In one hour he expects to serve $(1 - 4/13) \cdot 120 \approx 83.08$ children. His increase in idle time is approximately equal to $(\frac{4}{13} - 0.25) = \frac{3}{52}$ of an hour each hour (approximately 3.46 minutes per hour), during which time he would expect to serve $\frac{3}{52} \cdot 120 \approx 6.92$ children. So his "limited waiting room" strategy costs him 6.92 customers per hour on average. He should ask himself how this compares with the money he loses from shoplifting.

For the last part we need to compute the expected number in the queue, L_q. The queue length will be 1 or 2, with probabilities p_2 and p_3 respectively. Hence

$$L_q = 1 \cdot p_2 + 2 \cdot p_3 = 1 \cdot \frac{3}{13} + 2 \cdot \frac{3}{13} = \frac{9}{13} \approx 0.6923$$

children.

18.6.2 Multiple servers

Example 18.4. Example 18.3 is modified as follows. Fred Smith has hired an assistant to help him in his shop during the busiest hour. As before, Fred can serve a customer in 30 seconds, and his new assistant is equally fast on average. Whenever the shop contains at most one child a child will usually enter after 15 seconds on average, which is faster than before because now children expect to be served promptly if there are two servers, but this rate drops to one every 30 seconds if there are two children in the shop and to only one a minute if there are already 3 in the shop. Fred will allow this extra (fourth) child to enter the shop in comparison to Example 18.3, because he feels with two people at work he has a better chance of spotting a shoplifter, but this is his limit: no more than 4 children are allowed in the shop.

For what proportion of the day does Fred expect to be idle? How many customers will now be served per hour? If he makes an average of 15p gross profit per child and he pays his assistant £9 for his hour's work each school day, is Fred going to find this more profitable than when he limited the number of children to just 3 at most and didn't have an assistant?

The number of customers in the shop, n, is now between 0 and 4.

When $n = 1$, we have that $\mu_1 = 120$ children per hour are served by either Fred of his assistant, but when $n = 2, 3, 4$, we have $\mu_n = 240$ per hour. The values of the arrival rates are $\lambda_0 = \lambda_1 = 240$ per hour, $\lambda_2 = 120$ per hour, $\lambda_3 = 60$ per hour, and $\lambda_n = 0$ for $n = 4, 5, 6, \ldots$.

Substituting into the equations (18.1):

$$\begin{aligned} p_1 &= \frac{240}{120}\, p_0 = 2\, p_0 \\ p_2 &= \frac{240 \cdot 240}{120 \cdot 240}\, p_0 = 2\, p_0 \\ p_3 &= \frac{240 \cdot 240 \cdot 120}{120 \cdot 240^2}\, p_0 = p_0 \\ p_4 &= \frac{240 \cdot 240 \cdot 120 \cdot 60}{120 \cdot 240^3}\, p_0 = \frac{1}{4}\, p_0. \end{aligned}$$

Thus $p_0(1 + 2 + 2 + 1 + \frac{1}{4}) = 1$.

Hence $p_0 = 4/25$, $p_1 = 8/25$, $p_2 = 8/25$, $p_3 = 4/25$, and $p_3 = 1/25$.

Fred is idle whenever the number in the system is $n = 0$, and on average half the time when $n = 1$ (because when there is only one child in the shop Fred and the assistant will be equally likely to be serving the child), so that the expected proportion of time Fred is idle is $p_0 + \frac{1}{2}p_1 = 8/25$ of the hour. Hence the number of children Fred expects to serve during the hour is $(1 - 8/25) \cdot 120 \approx 81.6$ children and, since the assistant is expected to serve the same number on average, approximately 163.2 children will be served during the lunch hour.

This is a large increase on the corresponding figure, 83.08, in Example 18.3. The expected extra 80.12 children served contribute an expected $80.12 \cdot 0.15 = £12.02$ additional gross profit. Since the wages of the assistant are only £9, Fred can expect on average that the strategy of letting up to four children in the shop will generate an extra £3.02 profit per school day peak hour.

18.7 Exercises

Exercise 18.1.

(a) Calculate the sum of the 100 numbers $3, 3.5, 4, 4.5, \ldots, 52.5$.

(b) Calculate the sum of the infinite series $5 + 5/3 + 5/9 + 5/27 + 5/81 + \cdots$

(c) Calculate the sum of the infinite series $2 + 6(1/4) + 10(1/16) + 14(1/64) + \cdots$

Exercise 18.2. A TV repairman finds that the time spent on his jobs has an exponential distribution with mean value 30 minutes. If he repairs sets in the order in which they come

in, and if the arrival of sets is approximately Poisson distributed with an average rate of 10 per 8-hour day, how much idle time will he expect to have each day? What is the probability at any time that there will be two or more sets waiting to be repaired? If a set has just been brought in, how many sets on average will be ahead of it in his workshop?

Exercise 18.3. The island of St Michael's Mount near Penzance is connected to the mainland by a causeway across which it is possible to walk for a period of several hours, twice a day. At other times the causeway is covered by the tidal water, and visitors to the island have to be transported by a motor launch, or get very wet! The launch collects a "group" of passengers (either a single individual or two or more family members or friends — a group never contains people who are strangers to each other) from the mainland quayside, takes them to the island, and disembarks them. It next collects anybody who is waiting to return, and returns to the mainland, where it disembarks the returnees (if any). If there are no people waiting to return, it leaves immediately anyway. Note that for the trip back to the mainland, the launch may carry passengers or groups who don't know one another, subject only to health and safety capacity limits, and may even include people who had walked across the causeway when the tide was out.

If there is no group waiting at the mainland quayside for a passage to the island, then the launch will wait there until one comes, even if it is clear that there are people on the island waiting to return to the mainland. On average around 10 groups arrive at the quayside per hour, wishing to use the launch, and they pay £5 per group, although this is reduced to only £2 if, on arrival, there is at least one other group in front of them at the quayside waiting for the launch. Returning groups from the island pay nothing for the privilege.

The average time for a complete cycle is 4 minutes: picking up one group at the mainland, travelling to the island, collecting returners (if appropriate), travelling back to the mainland, and disembarking any returners.

Typically the launch operates twice a day (i.e., in two shifts) for around seven and a half hours each time, until the tide next recedes sufficiently for the causeway to become accessible. The launch operators pay the boatman £75 per shift, and the daily cost of fuelling, maintaining and insuring the launch is £80. How much profit can the launch owners expect to earn per day? What assumptions have you made? How reasonable are these assumptions?

Exercise 18.4. A bank employs three tellers. Each can serve a customer in 2 minutes and 24 seconds on average. Customers arrive individually and at random at a rate of 50 per hour, following a Poisson distribution. Answer the following questions about the steady state:

(a) Calculate p_0 and the probability there are n customers in the bank (for $n \geq 1$).

(b) For what proportion of the day will each teller be busy on average?

(c) Calculate the expected number of customers in the queue.

(d) If, for security reasons, at most 5 customers are allowed in the bank at any time, calculate the percentage decrease in the expected number of customers in the queue. (Assume that customers arriving when the bank is full go away and do not return.)

Exercise 18.5. Show that, in the case $s = 2$ and with independent arrival rates,

1. $p_0 = \dfrac{1-\rho}{1+\rho}$;
2. $L_q = \dfrac{2\rho^3}{(1-\rho^2)}$;
3. $W_q = \dfrac{\rho^2}{\mu(1-\rho^2)}$;
4. $W_S = \dfrac{1}{\mu(1-\rho^2)}$;
5. $L_S = \dfrac{2\rho}{(1-\rho^2)}$.

Exercise 18.6. Lorries arrive at an unloading bay according to a Poisson process with an average rate of 1 every 40 minutes. They can be unloaded at one of 10 bays, provided the bay is in service, and the unloading time is exponential with a mean of 1 hour. It costs £20 per hour to have a lorry idle in the depot, and the cost of having any specific bay in service is £30 per hour. What is the number of bays in service that will minimise the total cost per lorry unloaded?

Exercise 18.7. Ratty is a farm dog. His master keeps him because he is very good at catching rats if they enter the hen house, where the hens lay eggs. Ratty prefers to chase a rat and play with it before killing it, so that when there is a single rat in the hen house Ratty takes 5 minutes on average to (catch and) kill it. If there are two live rats in the hen house Ratty takes on average 3 minutes to kill the first of them, but if there are more than two rats running around in the hen house Ratty gets confused and each kill takes him 10 minutes. Rats enter the hen house randomly and independently at an average rate of once every 10 minutes, but some shared instinct means that there will never be more than four live rats in the hen house at any time. On the other hand there is no way out for a rat except as a dead rat!

(a) For what fraction of his time in the hen house is Ratty able to rest?

(b) Suppose a rat has just entered the hen house and observes with horror that Ratty is in the hen house and is currently occupied with another rat. On average how long can this rat expect to have to wait before starting to receive Ratty's attentions?

Exercise 18.8. Bill and Ben run an emergency drop-in dental clinic, and share the profits from it equally. Typically they can each treat a patient in 20 minutes. When prospective patients arrive at the entrance to the clinic they are more likely to balk and leave the clinic without waiting to get treatment if the clinic appears to be relatively full. Specifically, if there are n patients already in the clinic, either receiving treatment or in the waiting room, the entry rate into the clinic becomes $10 - 2n$ patients per hour, where $n \leq 5$. Whenever $n > 5$ all prospective patients will balk. Patients who receive treatment without having to wait until either Bill or Ben is free are charged £50 per treatment, but if they have to queue, even for one second, this charge is reduced to £20. In addition, half the patients buy on average £10 worth of dental-related goods, such as toothbrushes or mouthwash, the mark-up

on these goods being 100 percent. The sales of these goods are handled by the receptionist, who costs Bill and Ben a total of £100 per day in wages, national insurance and related costs. Note that the nominal operational day is six hours, but any patients still in the clinic at the end of the day will receive their completed treatment. The other costs of running the clinic, excluding Bills and Ben's fees, are £100 per day. Answer the following questions:

(a) Calculate the steady state probabilities that there are n patients in the clinic, for $0 \leq n \leq 5$.

(b) For how long in total during the nominal operational day can Bill expect to be free?

(c) On average how many patients will be treated at the clinic each day?

(d) What will be Bill's share of the daily pre-tax profit?

Bibliography

[AMO] Ahuja, R. K., Magnanti, T. L., Orlin, J. B. *Network flows.* Prentice Hall Inc, 1993.

[Ax] Sven Axsäter, Using the Deterministic EOQ Formula in Stochastic Inventory Control, *Management Science*, **42** (1996), 830-834.

[C] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. *Introduction to Algorithms.* Cambridge: MIT press. Third edition, 2009.

[HL] Hillier, F. S., Lieberman, G. J. *Introduction to Operations Research.* McGraw-Hill, 7th edition, 2005/9th edition, 2010.

[K] Kleinberg, J., Tardos, É. *Algorithm Design.* Pearson Education, 2006.

[Pa] Papadimitriou, Ch. *Computational Complexity.* Addison-Wesley, 1994.

[Pi] Pidd, M. *Tools for Thinking - Modelling in Management Science.* Wiley, 3rd edition, 2009.

[WA] Winston, W., Albright, S. *Practical Management Science*, Cengage Learning, 2011.

Appendix A

A.1 Derivation of batch-size and stock level for EOQ model with shortages

Recall that we needed to derive the quantities Q and S in order to minimise the function:

$$\textit{Cost per time: } C(Q,S) = \frac{Kd}{Q} + dc + \frac{hS^2}{2Q} + \frac{p(Q-S)^2}{2Q}.$$

To do this, we need to set the partial derivatives $\partial C(Q,S)/\partial Q$ and $\partial C(Q,S)/\partial S$ to 0. We compute the partial derivatives:

$$\begin{aligned}
\frac{\partial C(Q,S)}{\partial Q} &= -\frac{Kd}{Q^2} - \frac{hS^2}{2Q^2} + \frac{2p(Q-S)Q - p(Q-S)^2}{2Q^2} \\
\frac{\partial C(Q,S)}{\partial S} &= \frac{hS}{Q} - \frac{p(Q-S)}{Q}.
\end{aligned}$$

Setting them to zero yields the equations:

$$\begin{aligned}
-2Kd - hS^2 + 2p(Q-S)Q - p(Q-S)^2 &= 0 \qquad \text{(A.1)} \\
hS - p(Q-S) &= 0. \qquad \text{(A.2)}
\end{aligned}$$

From (A.2), we have $p(Q-S) = hS$, hence, if we replace hS for $p(Q-S)$ in (A.1), we obtain

$$-2Kd - hS^2 + 2hSQ - hS(Q-S) = 0.$$

After two cancellations, the previous equation reduces to

$$-2Kd + hSQ = 0. \qquad \text{(A.3)}$$

From (A.2), we get

$$Q = \frac{(h+p)}{p}S,$$

hence substituting for Q in (A.3), we obtain

$$-2Kd + \frac{h(h+p)}{p}S^2 = 0,$$

which gives the optimal value of S

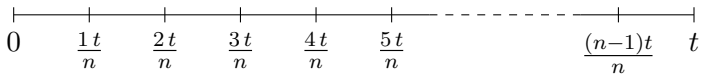
$$S^* = \sqrt{\frac{2dK}{h} \cdot \frac{p}{p+h}}.$$

Finally

$$Q^* = \frac{(h+p)}{p} S^* = \sqrt{\frac{2dK}{h} \cdot \frac{p+h}{p}}.$$

A.2 The Poisson process, Poisson distribution, and exponential distribution

Subdivide the time interval [0,t) into a large number n of subintervals of width t/n, as illustrated below.



If n is large enough, then by assumption (b) in Section 18.2.2, the probability of exactly one arrival occurring in any given time interval is "essentially" $\lambda t/n$ (to within terms of order smaller than $1/n$), and the probability of no arrivals is essentially $1 - \lambda t/n$. The *Poisson process* is the set of all arrival times, in the large-n limit.

The *Poisson distribution* governs the number of arrivals $X(t)$ occurring in time t. To compute it, note that "k arrivals in $[0, t)$" means that some k of the n subintervals each have exactly one arrival, and the other $n - k$ have no arrivals. By assumption (a), arrivals in each of the n intervals are independent, thus the probability of exactly k arrivals is given by the binomial distribution

$$f_n(k) = \binom{n}{k} \left(\frac{\lambda t}{n}\right)^k \left(1 - \frac{\lambda t}{n}\right)^{n-k}, \quad k = 0, 1, 2, 3, \ldots$$

Recalling that $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ and rearranging factors in the product above, it follows that:

$$\begin{aligned}
\mathbb{P}(X(t) = k) &= \lim_{n\to\infty} f_n(k) \\
&= \frac{(\lambda t)^k}{k!} \cdot \lim_{n\to\infty} \frac{n!}{(n-k)!} \frac{1}{n^k} \cdot \lim_{n\to\infty} \left(1 - \frac{\lambda t}{n}\right)^n \cdot \lim_{n\to\infty} \left(1 - \frac{\lambda t}{n}\right)^{-k} \\
&= \frac{(\lambda t)^k}{k!} e^{-\lambda t},
\end{aligned}$$

where the last equality comes from the fact that the first and third limits are 1, while the second is $e^{-\lambda t}$. This is the Poisson distribution function with rate λt. Setting $t = 1$ gives the familiar rate-λ Poisson distribution function $e^{-\lambda}\lambda^k/k!$.

We now show that the *exponential distribution* governs the interarrival time T. Condition on the event that there is an arrival at time τ, and, for any given t, consider the event that the next interarrival time T has $T > t$, i.e., the event that there is no arrival in the interval

$(\tau, \tau + t)$. By the memoryless property, this event is independent of there having been an arrival at time τ, thus its probability is simply the (unconditioned) probability of there being 0 events in the interval $(\tau, \tau + t)$. But the calculation above for the number of events in the interval $[0, t)$ applies equally to this interval (indeed to any length-t interval). Thus,

$$\mathbb{P}(T > t) = \mathbb{P}(\text{no event in } (\tau, \tau + t)) = \mathbb{P}(X(t) = 0) = e^{-\lambda t}.$$

Let $A(t)$ be the CDF of the interarrival time distribution. By definition, $A(t) = \mathbb{P}(T \leq t)$, so

$$A(t) = 1 - \mathbb{P}(T > t) = 1 - e^{-\lambda t}.$$

In essence we are now done: $A(t)$ is familiar as the CDF of an exponential distribution of rate λ. All properties of this distribution are well known, including the CDF and the mean, but for thoroughness, let us derive both.

By definition, the CDF $A(t)$ is the integral of the PDF $a(t)$, i.e., $A(t) = \int_0^t a(\tau) d\tau$, so the PDF is the derivative of the CDF: $a(t) = \frac{d}{dt} A(t)$. Here,

$$a(t) = \frac{d}{dt}(1 - e^{-\lambda t}) = \lambda e^{-\lambda t}.$$

The mean interarrival time is

$$\begin{aligned} \mathbb{E}[T] &= \int_0^\infty t\, a(t)\, dt = \int_0^\infty t\, \lambda e^{-\lambda t}\, dt = \frac{1}{\lambda} \int_0^\infty x e^{-x}\, dx \\ &= \frac{1}{\lambda} \left(-(1 + x) e^{-x} \right) \Big|_0^\infty = \frac{1}{\lambda}. \end{aligned}$$

A.3 Geometric progression

A *geometric progression* is a sequence of the form $1, a, a^2, a^3, \ldots\ a^n, \ldots$, where $a > 0$. The sum of the first n terms of the sequence, called a geometric *series*, is

$$S_n = 1 + a + a^2 + a^3 + \cdots + a^n.$$

Multiplying by $1 - a$ on both sides of the above equality, we obtain

$$(1 - a) S_n = (1 + a + a^2 + a^3 \cdots + a^n) - (a + a^2 + a^3 + \cdots a^n + a^{n+1}) = 1 - a^{n+1},$$

implying

$$S_n = \frac{1 - a^{n+1}}{1 - a}.$$

The sum of an infinite geometric series is $+\infty$ if $a \geq 1$, while if $a < 1$ it is:

$$1 + a + a^2 + a^3 + \cdots = \lim_{n \to +\infty} S_n = \lim_{n \to +\infty} \frac{1 - a^{n+1}}{1 - a} = \frac{1}{1 - a}. \tag{A.4}$$

A.4 Arithmetic-geometric progression

An *arithmetic geometric progression* is a sequence of the form $a, 2a^2, 3a^3, \dots\ na^n, \dots$, where $a > 0$. The sum of the first n terms of the sequence is

$$S_n = a + 2a^2 + 3a^3 + \cdots + na^n.$$

Multiplying by $1 - a$ on both sides of the above equality, we obtain

$$\begin{aligned}(1-a)S_n &= (a + 2a^2 + 3a^3 \cdots + na^n) - (a^2 + 2a^3 + 3a^4 + \cdots (n-1)a^n + na^{n+1}) \\ &= a + a^2 + a^3 + \cdots + a^n - na^{n+1},\end{aligned}$$

implying

$$S_n = \frac{a}{1-a}(1 + a + \cdots + a^{n-1}) - \frac{na^{n+1}}{1-a}.$$

Note that the term in parenthesis is a geometric series, therefore it equals $(1 - a^n)/(1 - a)$, and we obtain

$$S_n = \frac{a - a^{n+1}}{(1-a)^2} - \frac{na^{n+1}}{1-a}.$$

The sum of an infinite arithmetic-geometric series is $+\infty$ if $a \geq 1$, while if $a < 1$ it is:

$$a + 2a^2 + 3a^3 + \cdots = \lim_{n \to +\infty} S_n = \frac{a}{(1-a)^2}. \tag{A.5}$$