Stochastic Decision Theory III

APP MATH 3020 & 7090

✧ ✧ ✧

Giang Nguyen
(bringing together notes of
Joshua Ross, David Green, and many others)

✧ ✧ ✧

Discipline of Applied Mathematics
School of Mathematical Sciences
University of Adelaide

✧ ✧ ✧

Last updated October 25, 2018

Publishing Company, Adelaide, SA.

Printed in the World

The paper used in this publication may meet the minimum requirements of the American National Standard for Information Sciences — Permanence of Paper for Printed Library Materials, ANSI Z39.48–1984.

10 09 08 07 06 05 04 03 02 01 15 14 13 12 11 10 9

First edition: July, 2017

Contents

Preface

People make decisions everyday:

- whether to take an umbrella to work;
- to take an available park for their car or continue to search for a better one;
- which of several possible methods to implement to attempt to save a species from extinction; and,
- which people in the population to give a vaccine to.

All of these decisions are being made under uncertainty:

- there exists a certain chance of rain today;
- a certain chance all the remaining car parks are occupied;
- uncertainty about how many individuals of the species exist and how they will respond to each of the possible interventions; and,
- the actual dynamics of the infection and the uptake of the vaccine by the population.

This course will focus on formulating problems of this type in a mathematical framework and provide methods for making the best decision possible taking into account the uncertainty. Three topics will be covered:

- I. Stochastic Linear Programming;
- II. Markov Decision Chains (MDCs); and,
- III. Hidden Markov Chains (HMCs).

This course is interesting as it brings together techniques from optimisation and applied probability, and uses these to address problems which have all the features of real-world decision problems.

Chapter 1

Introduction and Revision

§1.1 Three exemplar problems

Let us consider three problems which highlight the topics of this course.

1. **Stochastic Linear Programming Example:**

   Consider a manufacturer which can make two products using two resources. The amount of each resource required to make each product, the demand for each product, the price of each product, and the supply and costs of each resource are provided in the Table 1.1.

	Resource 1	Resource 2	Demand	Price
Product 1	2	1	10	9
Product 2	3	2	10	12
Supply	50	40		
Cost	2	2		

Table 1.1: Example 1 data.

Given this information, in Optimisation and Operations Research II, we

would have formulated the *Linear Program*:

$$\begin{aligned} \text{maximise} \quad & z = 9x_1 + 12x_2 - 6x_1 - 10x_2 \\ \text{such that} \quad & x_1 \geq 10 \quad \text{—fixed demand \#1} \\ & x_2 \geq 10 \quad \text{—fixed demand \#2} \\ & 2x_1 + 3x_2 \leq 50 \\ & x_1 + 2x_2 \leq 40 \\ & x_1,\, x_2 \geq 0 \end{aligned}$$

where $x_i, i = 1,2$ is the number of product i produced.

However, if we consider the real-world scenario, it is typical for the manufacturer to be uncertain as to *demand* prior to producing the goods. For example, consider a shop and customers. Hence, $D_i, i = 1,2$ – the demand for product i – is really a random variable. Furthermore, other parts of the problem may also be similarly unknown, and stochastic.

Consequently, we might want to consider the "mathematical program":

$$\begin{aligned} \text{`maximise'} \quad & z = 9x_1 + 12x_2 - 6x_1 - 10x_2 \\ \text{such that} \quad & \text{“}x_1 \leq D_1\text{"} \\ & \text{“}x_2 \geq D_2\text{"} \\ & 2x_1 + 3x_2 \geq 50 \\ & x_1 + 2x_2 \leq 40 \\ & x_1,\, x_2 \geq 0 \end{aligned}$$

but note that z is now a random variable, so what does "maximise" mean in this context? Furthermore, what does "$x_i \geq D_i$" mean?

Given a realisation of (D_1, D_2), then we could choose decision variables (x_1, x_2) to maximise the realisation of z, but what about the situation where we wish / need to made a decision now, before the realisation?

We will consider this problem; we'll assume we know the distribution function of (D_1, D_2) (the random components), and that these random

variables are independent of the decision variables. We will focus on how to formulate so-called *deterministic equivalent problems*, which may (potentially) be solved using existing mathematical programming techniques.

2. **Markov Decision Chains Example:**

   Eleanor is looking for a parking space on their way to a concert. Each parking space is unoccupied with probability (w.p.) p independently of whether other parking spaces are occupied or not. She cannot see if other parking spaces are occupied (due to a thick fog!), and hence can only observe the occupancy of the parking space immediately in front of her car.

   If she parks s spaces from the entrance, she will miss the first s minutes of the concert, $s = 0, 1, \ldots$. If she passes the entrance without having parked, she will miss D minutes.

   What is the optimal policy to adopt? That is, if the driver finds a free park $\ell \in \{s, s-1, \ldots, 0\}$ spaces from the entrance, should she park there?

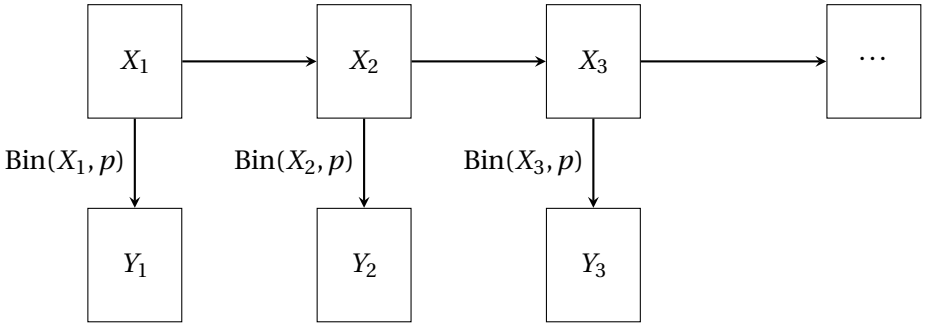
   Some of the key features of these problems are:

   - a Markov chain – here very simple, Discrete-time Markov chain (DTMC) with state space $\mathscr{S} = \{0, 1\}$, $t = s-$number of parking spaces until entrance$= 0, 1, \ldots, s+1$.
   - Actions – if full, no choice, but if empty, we may choose to park or not (state dependent).
   - Costs – upon parking at t, cost is $s - t$, for $t < s+1$, and costs D otherwise.

3. **Hidden Markov Chain Example:**

Consider a DTMC $\mathscr{X} = \{X_n, n = 0, 1, \ldots\}$ evolving in time, $n = 0, 1, \ldots$, and assume that at each time step we can only observe a partial observation, Y_n, of the process.

For example, at each time step we might have $Y_n \sim \text{Bin}(X_n, p)$.



This is a common situation in the real-world. For example, consider an epidemic – the spread of an infectious disease – where X_n is the actual number of people infected in a city on day n. Each of these individuals is only going to be recorded (known to government authorities) with some probability, say p; this corresponds to a probability of displaying symptoms and seeking medical attention.

In such a situation, an obvious question is, how many people are really infected in the city on each day, given I only observe a realisation of (Y_n)? We might additionally be interested in estimating the parameters of the underlying DTMC given these partial observations, or even wish to determine the best use of an antiviral. We will consider questions of this type.

§1.2 PROBABILITY

This section follows **very closely** the reference [**?**]. Another excellent reference is [**?**].

As we are dealing with random variables (rvs) and discrete-time Markov chains (DTMCs) in this course, we will spend a little bit of time reviewing the necessary elements of basic probability and DTMCs.

BASIC PROBABILITY ELEMENTS

Here we introduce the basic fundamental elements of probability theory.

Defn 1.1. *The **sample space** Ω of a random experiment is the set of all possible outcomes of the experiment.*

***Example* 1.** Tossing a coin: $\Omega = \{H, T\}$.
Rolling a dice: $\Omega = \{1, 2, \ldots, 6\}$.

Defn 1.2. *An **event** is a subset of the sample space Ω to which a probability can be assigned.*

***Example* 2.** Tossing a coin – getting a head: $\{H\}$.
Rolling a dice – getting greater than 3: $\{4, 5, 6\}$.

Defn 1.3. *A $\sigma-$algebra $\mathscr{F}$ on Ω is a collection of subsets of Ω that satisfies:*

1. $\Omega \in \mathscr{F}$,
2. *If* $A \in \mathscr{F}$ *then also* $A^c \in \mathbb{F}$,
3. *If* $A_1, A_2, \ldots \in \mathscr{F}$, *then* $\cup_n A_n \in \mathscr{F}$.

Defn 1.4. *A probability (or measure) P is a function which assigns a number between 0 and 1 to each event, and which satisfies the following rules:*

1. $0 \leq P(A) \leq 1$.
2. $P(\Omega) = 1$.
3. *For any sequence $A_1, A_2, \ldots$ of disjoint events we have*

$$P(\cup_i A_i) = \sum_i P(A_i).$$

For completeness, we note the following.

Defn 1.5. *The triple $(\Omega, \mathscr{F}, P)$ is called a **probability space**.*

How do we evaluate probabilities of particular events of interest? In other words, what are the basic properties of a probability? We now consider such basic properties.

Theorem 1.1. *Let A and B be events and P a probability. Then,*

1. $P(\emptyset) = 0$,
2. *if $A \subset B$, then $P(A) \leq P(B)$,*
3. $P(A^c) = 1 - P(A)$,
4. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.

Defn 1.6. *The **conditional probability** of A given B (with $P(B \neq 0)$) is defined as:*

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

***Example* 3.** Consider rolling 2 dice without looking. If someone informs you that the sum is 10, what is the probability that precisely one 6 was rolled?

Let B = event that sum is 10 = $\{(4,6),(5,5),(6,4)\}$ and let A = event that one 6 is rolled = $\{(1,6),(2,6),\ldots,(5,6),(6,1),\ldots,(6,5)\}$. Then we have $A \cap B = \{(4,6),(6,4)\}$. Therefore,

$$P(A|B) = \frac{2/36}{3/36} = \frac{2}{3},$$

as all outcomes equally likely.

Theorem 1.2. ***Product Rule**. Let $A_1,\ldots,A_n$ be a sequence of events with $P(A_1,\ldots,A_n) > 0$. Then,*

$$P(A_1,\ldots,A_n) =$$
$$P(A_1)P(A_2|A_1)P(A_3|A_1,A_2)\ldots P(A_n|A_1,\ldots,A_{n-1}).$$

Note, $P(A_1,\ldots,A_n) = P(A_1 \cap A_2 \cap \ldots \cap A_n)$.

***Example* 4.** Do you think 2 of us in this class share the same birthday (i.e. day of the year)? What do you think the probability is?

Let A_i be the event that the first i people have different birthdays, $i = 1,2,\ldots,n$, where n is the class size. Note that $A_n \subset A_{n-1} \subset \ldots A_2 \subset A_1$. Therefore $A_n = A_1 \cap A_2 \cap \ldots \cap A_n$, and thus by the product rule we have

$$P(A_n) = P(A_1)P(A_2|A_1)P(A_3|A_2)\ldots P(A_n|A_{n-1}).$$

Now, $P(A_k|A_{k-1}) = \frac{365-k+1}{365}$. Thus

$$P(A_n) = \frac{364 \times 363 \times \ldots \times (365-n+1)}{365 \times 365 \times \ldots \times 365}, \; n \geq 1.$$

Note, for $n = 23$ (approximately our class size) we have $P(A_{23}) < 1/2$ ($\approx 1/2$).

Theorem 1.3. ***Law of Total Probability***. *Let A be an event and let $B_1, B_2, \ldots, B_n$ be a partition of Ω. Then*

$$P(A) = \sum_{i=1}^{n} P(A|B_i)P(B_i).$$

Defn 1.7. *The events $A_1, A_2, \ldots,$ are said to be **independent** if for any k and any choice of distinct indices $i_1, \ldots, i_k$,*

$$P(A_{i_1} \cap A_{i_2} \cap \ldots \cap A_{i_k}) = P(A_{i_1})P(A_{i_2}) \ldots P(A_{i_k}).$$

PROBABILITY DISTRIBUTIONS

A formal definition of a rv is the following.

Defn 1.8. *A **random variable** is a function from the sample space Ω to $\mathbb{R}$.*

More precisely, we have the following definition.

Defn 1.9. *Let $(\Omega, \mathscr{F}, P)$ be a probability space. A **random variable** X is a numeric function on Ω that satisfies*

$$\{X \leq x\} \in \mathscr{F},$$

for all $x \in \mathbb{R}$. Here $\{X \leq x\}$ is an abbreviation of the set $\{\omega \in \Omega : X(\omega) \leq x\}$.

Let X be a rv. We will often desire the probabilities of events, such as $\{X = x\}$ and/or $\{a < X \leq b\}$. If we can specify all such probabilities involving X, then we say we have determined the probability distribution of X.

Defn 1.10. *The **cumulative distribution function** (cdf) of a random variable X is the function $F : \mathbb{R} \to [0,1]$ defined by*

$$F(x) = P(X \leq x), \qquad x \in \mathbb{R}.$$

DISCRETE DISTRIBUTIONS

Defn 1.11. *A rv X is said to have a **discrete distribution** if $P(X = x_i) > 0$, $i = 1, 2, \ldots$, such that $\sum_i P(X = x_i) = 1$. The **probability mass function** (pmf) of X is the function f defined by $f(x) = P(X = x)$.*

***Example* 5.** The *binomial distribution* models the number of successes from n trials, where success occurs wp p and all trials are independent. As a fixed example, think of the number of heads from n tosses of a fair coin, so $p = 1/2$. The binomial pmf is

$$f(x) = \binom{n}{x} p^x (1-p)^{n-x}, \; x \in \{0, 1, \ldots, n\}.$$

A plot of this pmf with $n = 10$ and $p = 1/2$ is in Figure 1.1.

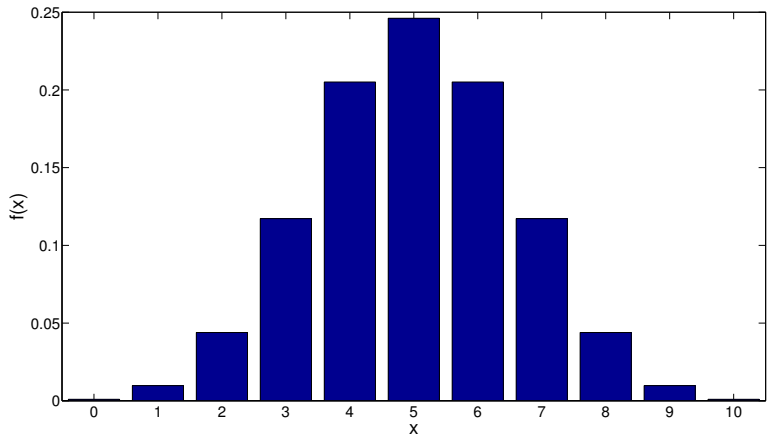


Figure 1.1: *Binomial pmf with* $n = 10$ *and* $p = 1/2$.

CONTINUOUS DISTRIBUTIONS

Defn 1.12. *A random variable X with cdf $F(x)$ is said to have a* ***continuous distribution*** *if there exists a positive function f, with total integral 1, such that for $a < b$,*

$$P(a < X \leq b) = F(b) - F(a) = \int_a^b f(x)dx.$$

The ***probability density function*** *(pdf) of X is the function f.*

***Example* 6.** An important continuous distribution is the exponential distribution. The holding time until a transition in a continuous-time Markov chain has an exponential distribution, where λ is the rate at which the chain leaves the current state. The pdf is

$$f(x) = \lambda e^{-\lambda x},\ x \geq 0$$

where λ is a fixed parameter. A plot of this pdf with $\lambda = 1/2$ is in Figure 1.2.

DISCRETE JOINT DISTRIBUTIONS

Defn 1.13 (Discrete Joint pmf)**.** *The joint pmf f of discrete rvs $X_1, X_2, \ldots, X_n$ is given by the function*

$$f(x_1, x_2, \ldots, x_n) = P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n).$$

Sometimes we may write $f_{X_1,\ldots,X_n}$ instead of f, to be emphatic about which random vector the pmf pertains to. Alternatively, if $\boldsymbol{X} = (X_1, \ldots, X_n)$ has been defined, we may write $f_{\boldsymbol{X}}$.

CONTINUOUS JOINT DISTRIBUTIONS

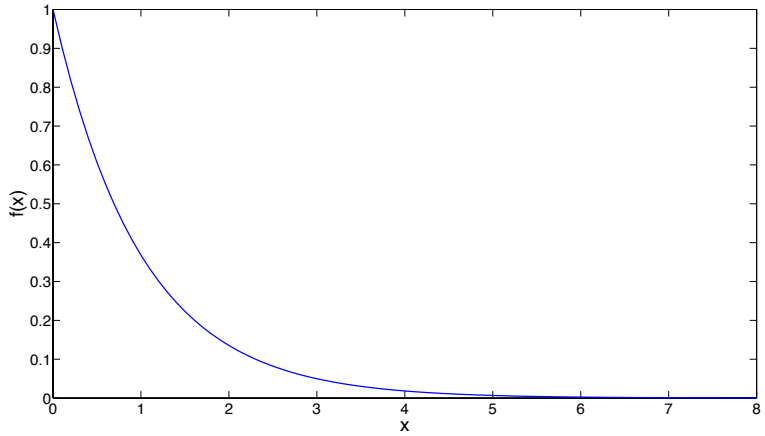


Figure 1.2: *Exponential PDF with* $\lambda = 1$.

Defn 1.14 (Continuous Joint pdf)**.** *Continuous random variables* $X_1, X_2, \ldots, X_n$ *are said to have a joint pdf* f *if*

$$\begin{aligned} &P(a_1 < X_1 \leq b_1, \ldots, a_n < X_n \leq b_n) \\ &= \int_{a_1}^{b_1} \ldots \int_{a_n}^{b_n} f(x_1, x_2, \ldots, x_n) dx_1 \ldots dx_n \end{aligned} \tag{1.1}$$

for all $a_1, \ldots, b_n$.

EXPECTATION

Defn 1.15 (Expectation of a Discrete Random Variable)**.** *Let X be a discrete rv with pmf f. The **expectation** (or expected value) of X, denoted $\mathbf{E}(X)$, is defined as*

$$\mathbf{E}(X) = \sum_x xP(X = x) = \sum_x xf(x),$$

provided that this sum is absolutely convergent.
(That is, $\sum_x |x| f(x) < \infty$.)

Example 7. Considering the binomial distribution again, we have

$$\mathbf{E}(X) = \sum_{x=0}^{n} x \binom{n}{x} p^x (1-p)^{n-x} = np.$$

One way to interpret the expectation is as a long-run average. If we repeatedly draw random numbers from the pmf f, then eventually the average will converge to $\mathbf{E}(X)$.

Defn 1.16 (Expectation of a Continuous Random Variable)**.** *Let X be a continuous rv with pdf f. The **expectation** (or expected value) of X, denoted as $\mathbf{E}(X)$, is defined as*

$$\mathbf{E}(X) = \int_{-\infty}^{\infty} xf(x)dx,$$

provided this integral is absolutely convergent.
(That is, $\int_{-\infty}^{\infty} |x| f(x)dx < \infty$.)

Example 8. Considering the exponential distribution again, we have

$$\mathbf{E}(X) = \int_0^{\infty} x\lambda e^{-\lambda x} dx = \frac{1}{\lambda}.$$

Note, if X is a rv then a function of X, such as X^2 or $\log(X)$, is also a rv.

Theorem 1.4 (Expectation of a Function of a rv)**.** *If X is discrete with pmf f, then for any real-valued function g,*

$$\mathbf{E}(g(X)) = \sum_x g(x) f(x).$$

Similarly, if X is continuous with pdf f, then

$$\mathbf{E}(g(X)) = \int_{-\infty}^{\infty} g(x) f(x)\, dx,$$

provided the sum/integral is absolutely convergent.

Theorem 1.5 (Properties of Expectation)**.** *For any real numbers a and b, and functions g and h,*

$$\mathbf{E}(aX + b) = a\mathbf{E}(X) + b,$$

$$\mathbf{E}(g(X) + h(X)) = \mathbf{E}(g(X)) + \mathbf{E}(h(X)).$$

Theorem 1.6 (Jensen's Inequality)**.** *Let $h(x)$ be a convex function and X a random variable. Then*

$$\mathbf{E}(h(X)) \geq h(\mathbf{E}(X)).$$

Proof. Assignment 1. □

§1.3 CONVEX AND QUASI-CONVEX FUNCTIONS, AND NONLINEAR OPTIMISATION

CONVEX COMBINATIONS AND SETS

Defn 1.17. *Given any finite collection of points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathbb{R}^n$, a point $\boldsymbol{z} \in \mathbb{R}^n$ is said to be a **convex combination** of the points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ if there is some $\lambda \in \mathbb{R}^m$ satisfying*

1. $\lambda_i \geq 0$, $i = 1, \ldots, m$
2. $\sum_{i=1}^m \lambda_i = 1$

such that $\boldsymbol{z} = \sum_{i=1}^m \lambda_i x_i$.

Defn 1.18. *A subset $\mathscr{S}$ of $\mathbb{R}^n$ is **convex** if the convex combination of any two points in $\mathscr{S}$ is also in $\mathscr{S}$. That is, for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathscr{S}$ and $\lambda \in (0,1)$, then*

$$\boldsymbol{x} = \lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2 \in \mathscr{S}.$$

***Example* 9.** The following Figure 1.3 gives an example of a convex set and a non-convex set.

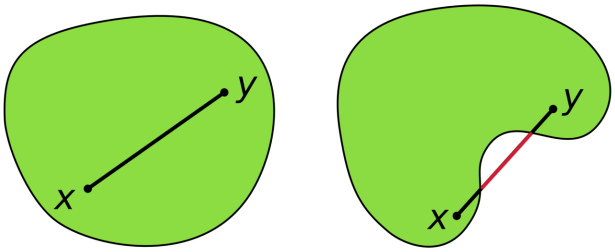


Figure 1.3: *A convex and non-convex set, respectively.*

CONCAVE AND CONVEX FUNCTIONS

Defn 1.19. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. The* ***subgraph*** *of f, denoted $sub(f)$, is the set*

$$sub(f) = \{(x, y) \in \mathscr{S} \times \mathbb{R} : f(x) \geq y\}.$$

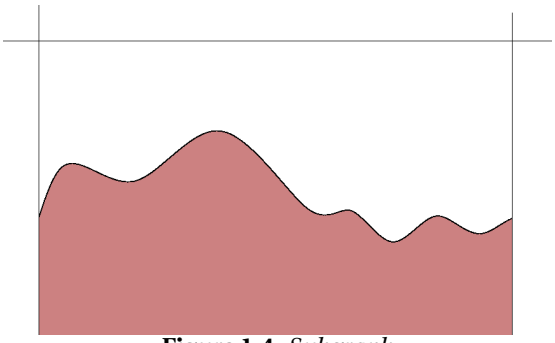


Figure 1.4: *Subgraph*

That is, the area lying below the graph of a function is its subgraph.

Defn 1.20. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. The **epigraph** of f, denoted $epi(f)$, is the set*

$$epi(f) = \left\{(x, y) \in \mathscr{S} \times \mathbb{R} : f(x) \leq y\right\}.$$

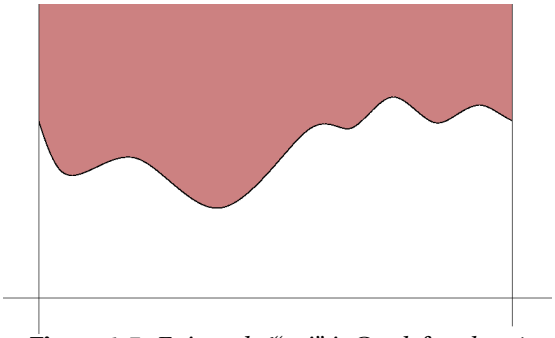


Figure 1.5: *Epigraph ("epi" is Greek for above)*

That is, the area lying above the graph of a function is its epigraph.

Defn 1.21. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **concave** on $\mathscr{S}$ if $sub(f)$ is a convex set.*

Defn 1.22. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **convex** on $\mathscr{S}$ if $epi(f)$ is a convex set.*

An alternative definition of concave and convex functions is available.

Defn 1.23. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **concave** on $\mathscr{S}$ if and only if $\forall\, x, y \in \mathscr{S}$ and $\lambda \in (0,1)$ we have*

$$f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda) f(y).$$

Defn 1.24. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **convex** on $\mathscr{S}$ if and only if $\forall\, x, y \in \mathscr{S}$ and $\lambda \in (0,1)$ we have*

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y).$$

We also have *strict* versions of these definitions.

Defn 1.25. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **strictly concave** on $\mathscr{S}$ if and only if $\forall\, x, y \in \mathscr{S}$, with $x \neq y$, and $\lambda \in (0,1)$ we have*

$$f(\lambda x + (1-\lambda)y) > \lambda f(x) + (1-\lambda) f(y).$$

Defn 1.26. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **strictly convex** on $\mathscr{S}$ if and only if $\forall\, x, y \in S$, with $x \neq y$, and $\lambda \in (0,1)$ we have*

$$f(\lambda x + (1-\lambda)y) < \lambda f(x) + (1-\lambda) f(y).$$

Theorem 1.7. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. Then*

1. *f is concave if and only if the function $(-f)$ is convex.*
2. *f is strictly concave if and only if the function $(-f)$ is strictly convex.*

This allows us to easily apply all results about concave functions to convex functions, and vice-versa.

There is also important consequences of convexity for optimisation, which can be essentially summarised as follows:

- if f is convex, then a local minimum is a global minimum; furthermore, it is either unique or there are infinitely many which are 'grouped together'.
- If the function f is strictly convex, then a local minimum is a global minimum and uniqueness is guaranteed.

Quasi-Concave and Quasi-Convex Functions

Whilst powerful, convexity is unfortunately a very restrictive assumption. It is often not satisfied in real-world applications. We will consider a slight generalisation of convexity, called quasi-convexity.

Defn 1.27. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. The **upper contour set** of f at $a \in \mathbb{R}$, denoted $U_f(a)$, is the set*

$$U_f(a) = \{x \in \mathscr{S} : f(x) \geq a\}.$$

Defn 1.28. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. The **lower contour set** of f, denoted $L_f(a)$, is the set*

$$L_f(a) = \{x \in \mathscr{S} : f(x) \leq a\}.$$

Defn 1.29. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **quasi-concave** on $\mathscr{S}$ if $U_f(a)$ is a convex set $\forall\, a \in \mathbb{R}$.*

Defn 1.30. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **quasi-convex** on $\mathscr{S}$ if $L_f(a)$ is a convex set $\forall\, a \in \mathbb{R}$.*

***Example* 10.** Below in figure (1.6) is an example, $f(x) = \sqrt{|x - 0.2|} + 0.2$, of a quasi-convex function which is not convex, and also an example of a non-quasi-convex function, $f(x) = 2(x^4 - x^2 + 0.35)$.

An example of a quasi-concave, but not concave function, is the Gaussian pdf.

Just as was the case for concave and convex functions, a relationship exists between the value of a function at two points and the value of the function at a convex combination:

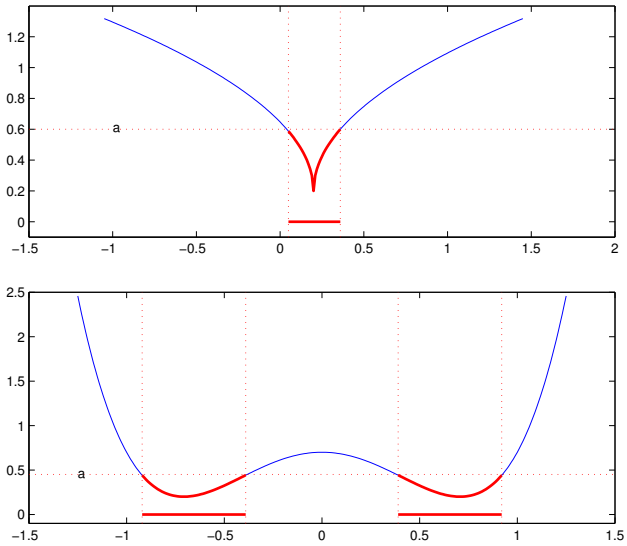


Figure 1.6: *Top: An example of a quasi-convex, but not convex, function. Bottom: a non-quasi-convex function, respectively.*

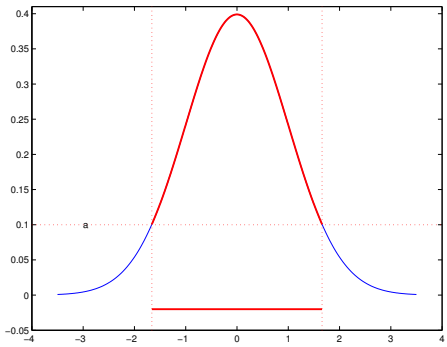


Figure 1.7: *The Gaussian pdf, which is quasi-concave but not concave.*

Theorem 1.8. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. Then the following statements are equivalent.*

1. *f is quasi-concave on $\mathscr{S}$.*
2. *for all $x, y \in \mathscr{S}$ and all $\lambda \in (0,1)$,*
$$f(x) \geq f(y) \implies f(\lambda x + (1-\lambda)y) \geq f(y).$$
3. *or all $x, y \in \mathscr{S}$ and all $\lambda \in (0,1)$,*
$$f(\lambda x + (1-\lambda)y) \geq \min\{f(x), f(y)\}.$$

Defn 1.31. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **strictly quasi-concave** on $\mathscr{S}$ if $\forall\, x, y \in \mathscr{S}$, with $x \neq y$, and $\lambda \in (0,1)$ we have*
$$f(\lambda x + (1-\lambda)y) > \min\{f(x), f(y)\}.$$

Defn 1.32. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a function. We say that f is **strictly quasi-convex** on $\mathscr{S}$ if $\forall\, x, y \in \mathscr{S}$, with $x \neq y$, and $\lambda \in (0,1)$ we have*
$$f(\lambda x + (1-\lambda)y) < \max\{f(x), f(y)\}.$$

Part of the flexibility of quasi-convexity over convexity is portrayed in the following theorem.

Theorem 1.9. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a quasi-concave function.*

1. *If $\phi : \mathbb{R} \to \mathbb{R}$ is an increasing function, then the composition $\phi \circ f$ is a quasi-concave function from $\mathscr{S} \to \mathbb{R}$.*
2. *In particular, any increasing transform of a concave function results in a quasi-concave function.*

The relationship between quasi-convexity and optimisation is summarised in the following theorem.

Theorem 1.10. *Let $\mathscr{S}$ be a convex subset of $\mathbb{R}^n$ and let $f : \mathscr{S} \to \mathbb{R}$ be a quasi-concave function. Then*

1. *Any local maximum of f is a global maximum of f.*
2. *The set $\arg\max\{f(x) : x \in \mathscr{S}\}$ of maximisers of f on $\mathscr{S}$ is either empty or a singleton.*

§1.4 Discrete-Time Markov Chains

A discrete-time Markov chain (DTMC) is a stochastic process – a sequence of random variables, evolving in time, corresponding to a sequence of random variables $X_n \in S$, where the random variable X_n is the state of the Markov chain at time n, and $\mathscr{S}$ is the state space of all values the Markov chain may adopt. A DTMC evolves in discrete time steps, $n = 0, 1, 2, \ldots$.

The use of *chain* (as opposed to 'process') implies that X_n is a discrete random variable.

Defn 1.33. *$\mathscr{X} = \{X_n, n \geq 0\}$ is called a discrete-time Markov chain provided that for all $x_0, x_1, \ldots, x_n, j \in S$ and $n \in \mathbb{N}$*

$$\begin{aligned} &\Pr(X_{n+1} = j | X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n) \\ &= \Pr(X_{n+1} = j | X_n = x_n) \quad \text{– Markov property,} \end{aligned}$$

with $\mathscr{S}$ being referred to as the state space, consisting of all values the Markov chain may adopt.

We will assume $\mathscr{S}$ is countable (and often furthermore finite) so that it can be put in one-to-one correspondence with a subset of the integers $S \subseteq \mathbf{Z}$.

Given that the chain is current in state i, say $X_n = i$, then we specify the probabilities of it being in state j at the next time step $n+1$:

$$\Pr(X_{n+1} = j | X_n = i) = p_{ij}(n).$$

Note that these probabilities depend on three quantities: the current state, i; the future state, j; and the time step n.

We will focus on *time-homogeneous* Markov chains where the transition probabilities only depend upon i and j:

$$\Pr(X_{n+1} = j | X_n = i) = p_{ij} = \Pr(X_1 = j | X_0 = i).$$

Considering these probabilities for each $i, j \in S$, we can form a (one-step) *transition (probability) matrix*

$$P = (p_{ij})$$

of size $|S| \times |S|$.

***Example* 11** (Tossing a coin n times)**.** Let X_n be the number of heads after n tosses, $n = 0, 1, \ldots, T$. Then we can form a one-step probability transition

matrix P as follows

$$P = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & \ddots & 0 \\ 0 & 0 & \frac{1}{2} & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix},$$

which is of dimension $|S| \times |S|$ where $S = \{0, 1, \dots, N\}$ being the set of possible numbers of heads from N coin tosses.

We now specify a (row) vector

$$\boldsymbol{p}(n) = (\Pr(X_n = 0), \Pr(X_n = 1), \dots, \Pr(X_n = N)),$$

specifying the pmf of the DTMC at time step n. Then, the jth entry of the vector $\boldsymbol{p}(n+1)$ is given by

$$p_j(n+1) = \Pr(X_{n+1} = j) = \sum_i p_i(n) p_{ij} = [\boldsymbol{p}(n)P]_j$$

and hence $\boldsymbol{p}(n+1) = \boldsymbol{p}(n)P$. This specifies the evolution of the DTMC. It follows that

$$\boldsymbol{p}(n) = \boldsymbol{p}(0)P^n,$$

where $\boldsymbol{p}(0)$ is the initial distribution of the DTMC.

***Example* 12.** So, considering our coin example again and assuming we start at time $n = 0$, then we have

$$\boldsymbol{p}(0) = (1, 0, \dots, 0) = \boldsymbol{e}_1^{N+1}$$

where $\boldsymbol{e}_b^a$ is a (row) vector of length a with a 1 in the bth entry and 0s elsewhere. Now, at time $n = 1$ we have

$$\boldsymbol{p}(1) = \boldsymbol{p}(0)P = \boldsymbol{e}_1^{N+1} \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & \ddots & 0 \\ 0 & 0 & \frac{1}{2} & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} = \left(\frac{1}{2}, \frac{1}{2}, 0, \dots, 0\right).$$

What do you think $\boldsymbol{p}(N)$ would look like, for $N = 10$? (Check!)

CLASSIFICATION OF STATES

Defn 1.34. *If states i and j are such that $[P^n]_{ij} > 0$ for some $n > 0$, we say that i* ***has access to*** *j and write $i \to j$.*

Defn 1.35. *We say that i and j* ***communicate*** *if $i \to j$ and $j \to i$. Then, we write $i \leftrightarrow j$.*

The relation $i \leftrightarrow j$ is an *equivalence relation*:

- $i \leftrightarrow i$
- $i \leftrightarrow j \iff j \leftrightarrow i$
- $i \leftrightarrow j, j \leftrightarrow k \iff i \leftrightarrow k$.

We can therefore divide $\mathscr{S}$ into equivalence (or communicating) classes such that states of a communicating class communicate with all other states within that class but not with any state outside that class.

Defn 1.36. *If there is only one communicating class ($= S$), the DTMC is said to be* ***irreducible****.*

Defn 1.37. *A communicating class is classified as* ***transient*** *if it is possible for the process to leave that class so that it can never return.*

On the other hand, a communicating class is classified as ***recurrent*** *if once the process enters that class, it can never leave.*

***Example* 13** (Toss a coin n times)**.** In the coin tossing example, we can draw a state diagram (see Figure 1.8) and use the notation of access to see what type of states and communicating classes we have.

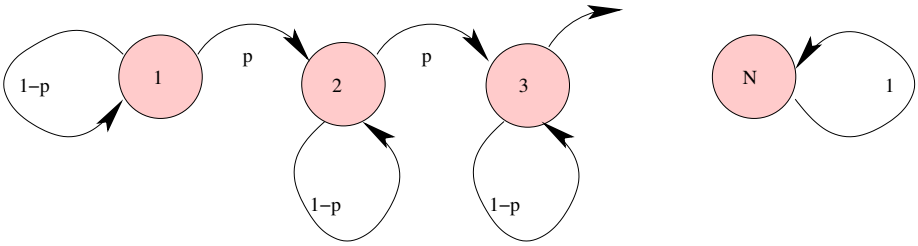


Figure 1.8: *State diagram for the coin tossing example.*

By Definition 1.37, each of the states $\{0,1,\ldots,N-1\}$ is their own communicating classes which are transient; the state N, on the other hand, is a communicating class that is recurrent or absorbing.

Equilibrium Behaviour of DTMCs

We can easily show that the probability entries $p_{ij}^{(m)}$ of the m-step transition probability matrix $P^{(m)}$, for $m \geq 1$, are given by

$$p_{ij}^{(m)} = \sum_{k \in S} p_{ik}^{(m-1)} p_{kj} \quad \text{for } i, j \in S. \tag{1.2}$$

For irreducible finite-state DTMCs, there is an **equilibrium behaviour**, that is,

$$\lim_{m \to \infty} p_{ij}^{(m)} = \pi_j. \tag{1.3}$$

In other words, the probability of being in state j after many steps converges to some constant value π_j and hence is independent of the initial state i. (Note that this property does not always occur with all Markov Chains that are not irreducible finite-state.)

Equation (1.2) and the assumption (1.3) implies that

$$\pi_j = \sum_{k \in S} \pi_k \, p_{kj}, \quad \text{for all } j \in S. \tag{1.4}$$

Equations (1.4) are known as the **equilibrium equations** for a Markov Chain and can be written in vector/matrix form as

$$\boldsymbol{\pi} = \boldsymbol{\pi} P, \tag{1.5}$$

where the row vector $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$. If in addition, $\boldsymbol{\pi}$ can be scaled such that $\sum_i \pi_i = 1$, then $\boldsymbol{\pi}$ is known as the **equilibrium probability distribution** of the DTMC, which, as can be seen from (1.5), is a left-eigenvector of P associated with an eigenvalue of 1.

***Example* 14.** Consider an irreducible four-state DTMC defined on $\mathscr{S} = \{0,1,2,3\}$ with the following transition probability matrix:

$$P = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & 0 & \frac{3}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \frac{3}{4} \\ 0 & \frac{1}{4} & \frac{3}{4} & 0 \end{pmatrix}.$$

The equilibrium probability equations $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbb{P}$ are

$$\pi_0 = \frac{1}{2}\pi_0 + \frac{1}{4}\pi_1 \quad \Rightarrow \quad \pi_0 = \frac{1}{2}\pi_1, \tag{1.6}$$

$$\pi_1 = \frac{1}{2}\pi_0 + \frac{1}{4}\pi_2 + \frac{1}{4}\pi_3, \tag{1.7}$$

$$\pi_2 = \frac{3}{4}\pi_1 + \frac{3}{4}\pi_3, \tag{1.8}$$

$$\pi_3 = \frac{3}{4}\pi_2. \tag{1.9}$$

Substituting (1.9) into (1.8), we have

$$\pi_2 = \frac{3}{4}\pi_1 + \frac{3}{4}\frac{3}{4}\pi_2 \quad \Rightarrow \quad \pi_2 = \frac{12}{7}\pi_1. \tag{1.10}$$

Similarly, substituting (1.10) into (1.9), we have

$$\pi_3 = \frac{3}{4}\pi_2 = \frac{3}{4}\frac{12}{7}\pi_1 = \frac{9}{7}\pi_1. \tag{1.11}$$

Then using the fact that $\sum_{i=0}^{3}\pi_i = 1$, we obtain

$$\boldsymbol{\pi} = (\pi_0, \pi_1, \pi_2, \pi_3) = \frac{14}{63}\left(\frac{1}{2}, 1, \frac{12}{7}, \frac{9}{7}\right) = \left(\frac{7}{63}, \frac{14}{63}, \frac{24}{63}, \frac{18}{63}\right).$$

Defn 1.38. *If a set of states C is such that*

$$\sum_{j \in C} p_{ij} = 1$$

for all $i \in C$, then C is called a ***closed set****.*

Defn 1.39. *A state i is called an* ***absorbing state*** *if $\{i\}$ is closed.*

Defn 1.40 (Absorbing class)**.** *A class C is called an* ***absorbing class*** *if C is closed.*

An irreducible DTMC effectively has a single absorbing class.

§1.5 LINEAR PROGRAMMING

In section I of the course, on Stochastic Linear Programming, we will formulate so-called *deterministic equivalent problems* (DEPs) of stochastic linear programs. Such DEPs are commonly general (nonlinear) mathematical programs, and occasionally will be linear programs. For this reason, we review (briefly) **Linear Programming**.

A mathematical programming problem (or mathematical program):

$$\begin{aligned} \text{minimise (or maximise)} \quad & z = g_0(x) \\ \text{such that} \quad & g_i(x) = (\text{or } \leq \text{ or } \geq)\, b_i,\ i = 1,\ldots,m \\ & x \in X \subset \mathbb{R}^n \end{aligned}$$

is called a **linear program** if $g_0(x) = c^\top x$ and $\{g_i(x)\} \equiv Ax$. That is, the objective function and constraints are all linear.

All linear programs can be expressed in the *standard form*:

$$\begin{aligned} \text{minimise} \quad & z = \boldsymbol{c}^\top \boldsymbol{x} \\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{aligned}$$

where $A \in \mathbb{R}^{m\times n}$, $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{b} \in \mathbb{R}^m$, rank$(A) = m \leq n$ (not essential) and $b_i \geq 0$ for $i = 1,\ldots,m$.

***Example* 15** (A scheduling problem)**.** A manufacturing company makes three types of items: desks, chairs, and bed-frames, using metal and wood.

A single desk requires 2 hours of labour, 1 unit of metal, 3 units of wood. One chair requires 1 hour of labour, 1 unit of metal, and 3 units of wood. Making one bed frame requires 2 hours of labour, 1 unit of metal and 4 units of wood. In a given time period, there are 225 hours of labour available, 117 units of metal, and 420 units of wood. The profit on one desk is \$13, one chair is \$12, and one bed frame is \$17.

The company wants a manufacturing schedule designed, which maximises profits without violating any constraint on resource availability during that time period.

To formulate the problem, we start by reading the problem statement carefully and then tabulating the data. The data for this example is listed in Table 1.2.

	Labour (hrs)	Metal	Wood	Profit
Desk	2	1	3	13
Chair	1	1	3	12
Bedframe	2	1	4	17
Total	225	117	420	

Table 1.2: Example 15 data

Then, we identify **decision variables**, which in this case are x_1, x_2, and x_3, where x_1 is the number of desks, x_2 the number of chairs, and x_3 the number of bed frames, made per time period. Next, we need to determine the **objective function**, which here is to maximise the profit, z, (in dollars); so, we have:

$$\max\ z = 13x_1 + 12x_2 + 17x_3.$$

The constraints are

$$\begin{aligned} 2x_1 + x_2 + 2x_3 &\leq 225 \quad \text{(labour)} \\ x_1 + x_2 + x_3 &\leq 117 \quad \text{(metal)} \\ 3x_1 + 3x_2 + 4x_3 &\leq 420 \quad \text{(wood)} \\ x_1, x_2, x_3 &\geq 0. \end{aligned}$$

The feasible region for this LP is depicted in Figure 1.9

Should we include any other constraints on the x_i's? If integer constraints were added (whole hours or whole units of base products), it would become an Integer Linear Program (ILP).

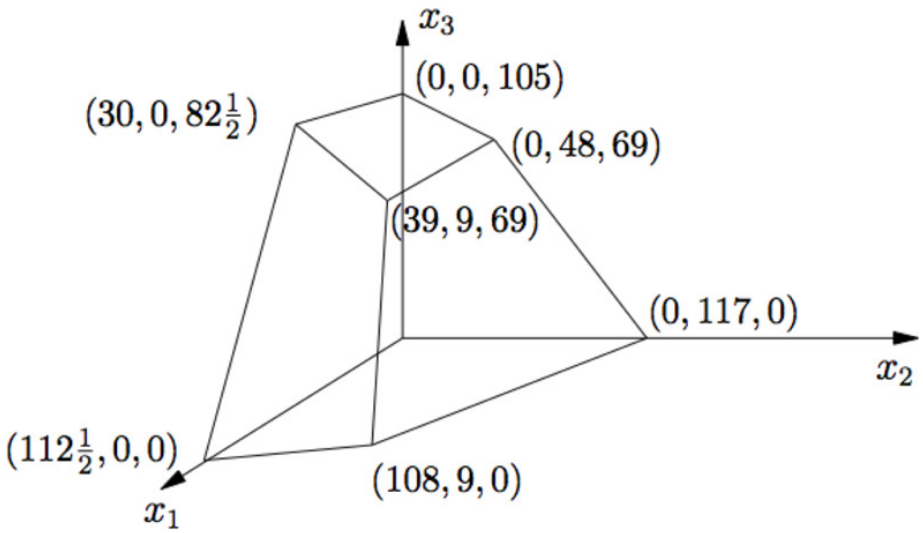


Figure 1.9: *Feasible region for the LP in Example 15.*

In summary, the *linear program* (LP) is

$$\begin{aligned} \max \quad & z = 13x_1 + 12x_2 + 17x_3 \\ \text{subject to} \quad & 2x_1 + x_2 + 2x_3 \leq 225 \\ & x_1 + x_2 + x_3 \leq 117 \\ & 3x_1 + 3x_2 + 4x_3 \leq 420 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

To write in *standard form,* we add **slack variables** $x_4, x_5, x_6 \geq 0$ as follows

$$\begin{aligned} \min \quad & -z = -13x_1 - 12x_2 - 17x_3 \\ \text{subject to} \quad & 2x_1 + x_2 + 2x_3 + x_4 = 225 \\ & x_1 + x_2 + x_3 + x_5 = 117 \\ & 3x_1 + 3x_2 + 4x_3 + x_6 = 420 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0. \end{aligned}$$

The variables x_4, x_5 and x_6 are known as slack variables because they actually take up the slack to convert an inequality into an equality.

We will revisit this example later to see a solution of the above LP.

Defn 1.41. *Variables x satisfying the constraints*

$$A\boldsymbol{x} = \boldsymbol{b}, \quad \boldsymbol{x} \geq \boldsymbol{0}$$

*are called **feasible solutions**.*

Defn 1.42. *Suppose rank$(A) = m$, then there exists m columns of A which are linearly independent, say the first m columns. Define*

$$B := (A_{:,1}, A_{:,2}, \ldots, A_{:m}),$$

*then B is non-singular. We say B is a **basis**.*

Defn 1.43. *If B is a basis, then the linear system*

$$B\boldsymbol{x}_B = \boldsymbol{b}$$

has a unique solution, $\boldsymbol{x}_B = B^{-1}\boldsymbol{b}$. Let $\boldsymbol{x} = (\boldsymbol{x}_B, \boldsymbol{0})$, then $\boldsymbol{x}$ satisfies

$$A\boldsymbol{x} = \boldsymbol{b},$$

*and $\boldsymbol{x}$ is called a **basic solution** with respect to B.*

*Furthermore, if $\boldsymbol{x}_B \geq \boldsymbol{0}$ it is called a **basic feasible solution**.*

*The $\boldsymbol{x}_B$ are called **basic variables**; the other variables corresponding to the 0s are called **non-basic variables**.*

***Example* 16.** Consider the linear program

$$\begin{aligned} \text{maximise} \quad & z = 9x_1 + 12x_2 - 6x_1 - 10x_2 \\ \text{such that} \quad & x_1 \leq 10 \\ & x_2 \leq 10 \\ & 2x_1 + 3x_2 \leq 50 \\ & x_1 + 2x_2 \leq 40 \\ & x_1, x_2 \geq 0 \end{aligned}$$

where $x_i, i = 1,2$ is the number of product i produced. In standard form we have

$$\begin{aligned} \text{minimise} \quad & z = -3x_1 - 2x_2 \\ \text{such that} \quad & x_1 + x_3 = 10 \\ & x_2 + x_4 = 10 \\ & 2x_1 + 3x_2 + x_5 = 50 \\ & x_1 + 2x_2 + x_6 = 40 \\ & x_1, x_2, \ldots, x_6 \geq 0. \end{aligned}$$

Here,

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 2 & 3 & 0 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \end{pmatrix},$$

and $\boldsymbol{b} = (10, 10, 50, 40)^\top$. By inspection, we can choose

$$B = (A_{:,3}, A_{:,4}, A_{:,5}, A_{:,6}) = I_{4\times 4},$$

as a basis, and hence $\boldsymbol{x} = (0, 0, 10, 10, 50, 40)^\top$ is a **basic feasible solution**; $\{x_3, x_4, x_5, x_6\}$ is the set of **basic variables**.

Defn 1.44. *The set of all the feasible solutions is called the* ***feasible region****.*

There are three possible cases for the feasible region:

- Empty set – no solution;
- Unbounded convex set (an infinite collection of solutions); or,
- A **bounded** convex polytope (convex bounded set, or convex hull) – bounded collection of solutions.

Defn 1.45. *Any point z in a convex set $\mathscr{S}$ which cannot be expressed as $z = \lambda x + (1-\lambda)y$ for $x, y \in S$ and $\lambda \in (0,1)$ is called an* ***extreme point****.*

The ***vertices*** *of a convex set are called* ***extreme points****.*

Theorem 1.11 (Optimum solution of an LP)**.** *An optimum of a linear objective function $z = \boldsymbol{c}^\top \boldsymbol{x}$, with convex polytope feasible region, occurs at an extreme point.*

Theorem 1.12. *A point in the feasible region is an extreme point if and only if it is a basic feasible solution.*

The **Simplex Method** is a procedure to solve linear programs in standard form. Essentially, it transfers from one basis (corresponding to a basic feasible solution) to another basis (also corresponding to a basic feasible solution) such that the objective function value never increases.

MATLAB LINPROG.M

In MATLAB, the function `linprog.m` allows one to solve linear programming problems. Please familiarise yourself with this algorithm; it will be assumed that you can solve linear programming problems using `linprog.m`.

To begin with, Matlab has a help command which enables you to find out about all the inbuilt commands. For example the command `linprog.m` can be investigated at the prompt as follows.

```
>> help linprog

 linprog Linear programming.

    X = linprog(f,A,b) attempts to solve the linear programming problem:

             min f'*x    subject to:   A*x <= b
             x
```

```
X = linprog(f,A,b,Aeq,beq) solves the problem above while additionally
satisfying the equality constraints Aeq*x = beq.

X = linprog(f,A,b,Aeq,beq,LB,UB) defines a set of lower and upper
bounds on the design variables, X, so that the solution is in
the range LB <= X <= UB. Use empty matrices for LB and UB
if no bounds exist. Set LB(i) = -Inf if X(i) is unbounded below;
set UB(i) = Inf if X(i) is unbounded above.

X = linprog(f,A,b,Aeq,beq,LB,UB,X0) sets the starting point to X0. This
option is only available with the active-set algorithm. The default
interior point algorithm will ignore any non-empty starting point.
```

***Example* 17.** For Example 15, using the original inequality form of the LP but rewriting it as a minimisation problem as

$$\min \quad -z = -13x_1 - 12x_2 - 17x_3 = \boldsymbol{c}^\top \boldsymbol{x} = (-13,-12,-17)\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix},$$

in Matlab we do the following

```
>> f=[-13,-12,-17];
>> A=[2 1 2 ; 1 1 1; 3 3 4]
>> b=[225 ; 117 ; 420]
>> LB=[0;0;0]
>> [x,fval]=linprog(f,A,b,[],[],LB)
Optimization terminated.
x =
   30.0000
    0.0000
   82.5000
```

```
fval =
  -1.7925e+03
```

This means we obtain the optimal value $z^* = 1792.5$ with

$$\boldsymbol{x} = (x_1, x_2, x_3)^\top = (30, 0, 82.5)^\top.$$

Also for Example 15, we could use the standard equality form of the LP, which is rewritten as a minimisation problem. In that case, in MATLAB we do the following

```
>> f=[-13,-12,-17,0,0,0];
>> Aeq=[2 1 2 1 0 0 ; 1 1 1 0 1 0 ; 3 3 4 0 0 1]
>> beq=[225 ; 117 ; 420]
>> LB=[0;0;0;0;0;0]
>> [x,fval]=linprog(f,[],[],Aeq,beq,LB)
Optimization terminated.

x =
   30.0000
    0.0000
   82.5000
    0.0000
    4.5000
    0.0000

fval =
  -1.7925e+03
```

Once again, we obtain $z^* = 1792.5$, but with the extended version

$$\boldsymbol{x} = (30, 0, 82.5, 0, 4.5, 0)^\top.$$

***Example* 18.** For Example 16, using the original inequality form of the LP but re-writing as a minimisation problem as

$$\min \quad -z = -3x_1 - 2x_2 = \boldsymbol{c}^\top \boldsymbol{x} = (-3,-2)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

in MATLAB we do the following

```
>> f=[-3;-2];
>> A=[1 0;0 1;2 3;1 2]
>> b=[10;10;50;40]
>> LB=[0;0]
>> [x,fval]=linprog(f,A,b,[],[],LB)
Optimization terminated.

x =
      10
      10

fval =
     -50
```

So we get the optimal value $z^* = 50$ with

$$\boldsymbol{x} = (x_1, x_2)^\top = (10,10)^\top.$$

Also for Example 16, we could use the standard equality form of the LP, which is also rewritten as a minimisation problem, so that in MATLAB we do the following

```
>> f=[-3;-2;0;0;0;0];
>> Aeq=[1 0 1 0 0 0;0 1 0 1 0 0;2 3 0 0 1 0;1 2 0 0 0 1]
>> beq=[10;10;50;40]
>> LB=[0;0;0;0;0;0]
>> [x,fval]=linprog(f,[],[],Aeq,beq,LB)
```

```
Optimization terminated.

x =
   10.0000
   10.0000
    0.0000
    0.0000
    0.0000
    5.0000

fval =
  -50.0000
```

We again get $z^* = 50$ but with $\boldsymbol{x} = (10, 10, 0, 0, 0, 5)^\top$.

DUALITY THEORY

Finally, we revise the **dual** of a linear program and **duality** theory. Given the linear program

$$\begin{aligned} (P)\ \text{minimise} \quad & z = \boldsymbol{c}^\top \boldsymbol{x} \\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{aligned}$$

then the *dual* is

$$\begin{aligned} (D)\ \text{maximise} \quad & \boldsymbol{w} = \boldsymbol{y}^\top \boldsymbol{b} \\ \text{subject to} \quad & A^\top \boldsymbol{y} \leq \boldsymbol{c} \\ & \boldsymbol{y}\ \text{free}. \end{aligned}$$

Why is this of interest? Recall the following theorem, which shows that a feasible solution of (D), if exists, would provide a **lower bound** for a feasible solution of (P).

Theorem 1.13. *For a feasible solution of (P) with value z, and a feasible solution of (D) with value w, we have*

$$z \geq w \quad \textit{(weak duality)}.$$

Theorem 1.14. *If (P) has an optimal solution then (D) has an optimal solution and*

$$\min z = \max w \quad \textit{(strong duality)}.$$

Hence, if (P) has no optimal solution ($z \to -\infty$) then (D) cannot have any feasible solution, and if (D) has no optimal solution ($w \to \infty$) then (P) cannot have any feasible solution.

Additional reading: The book Operations Research: An Introduction, by Taha [**?**] (available in UoA library, Chapters 2–4), and Matt Roughan's online lecture notes for Optimisation and Operations Research II (available online).

Generalised Primal/Dual pairs

	Primal (Dual) $\max z = \sum_{j=1}^{n} c_j x_j + z_0$	**Dual (Primal)** $\min w = \sum_{i=1}^{m} y_i b_i + z_0$
1.	$\sum_{j=1}^{n} a_{ij} x_j = b_i$	y_i free
2.	$\sum_{j=1}^{n} a_{ij} x_j \leq b_i$	$y_i \geq 0$
3.	$\sum_{j=1}^{n} a_{ij} x_j \geq b_i$	$y_i \leq 0$
4.	$x_j \geq 0$	$\sum_{i=1}^{m} y_i a_{ij} \geq c_j$
5.	$x_j \leq 0$	$\sum_{i=1}^{m} y_i a_{ij} \leq c_j$
6.	x_j free	$\sum_{i=1}^{m} y_i a_{ij} = c_j$

Chapter 2

Stochastic Linear Programming

This section follows **very, very closely** the reference [**?**]. Another very good reference is [**?**].

§2.1 Motivation

As we have seen in Optimisation and Operations Research II, many real-world decision problems can be modelled as *linear programs*. That is, as a mathematical program of the form:

$$\begin{aligned} \text{minimise} \quad & z = \boldsymbol{c}^\top \boldsymbol{x} \\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{x} \geq \boldsymbol{0}. \end{aligned} \tag{2.1}$$

In (2.1), the *cost* coefficients (c_j), the *constraint* coefficients (a_{ij}), and the *demand/supply* coefficients (b_i) are all assumed to have fixed known values. Furthermore, the model (2.1) is only appropriate when the objective function and constraints are (at least close to) linear in the decision variables (x_i).

More generally, we have mathematical programs of the form:

$$\begin{aligned} \text{minimise} \quad & z = g_0(\boldsymbol{x}) \\ \text{such that} \quad & g_i(\boldsymbol{x}) \leq 0,\ i = 1,\ldots,m \\ & \boldsymbol{x} \in \mathscr{X} \subset \mathbb{R}^n \end{aligned} \tag{2.2}$$

The set $\mathscr{X}$ and functions $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 0,\ldots,m$, are defined by the modelling process (i.e., they are problem specific). Depending upon the properties of the set $\mathscr{X}$ and functions (g_i), the problem (2.2) has different names:

1. **linear**, if the set $\mathscr{X}$ is a convex polytope (bounded or unbounded) and the functions g_i's are linear;

2. **nonlinear**, if at least one of the functions g_i's is nonlinear, or $\mathscr{X}$ is not a convex polytope. Nonlinear programs can be further classified as:

   a) **convex**, if $\mathscr{X} \cap \{x : g_i(x) \leq 0, i = 1,\ldots,m\}$ is a convex set and g_0 is a convex function — for example, if g_i's are all convex functions and $\mathscr{X}$ is a convex set; or,

   b) **nonconvex**, if either $\mathscr{X} \cap \{x : g_i(x) \leq 0, i = 1,\ldots,m\}$ is not a convex set or the objective function g_0 is not convex.

In many modelling scenarios, it is unreasonable to expect the entries of the vectors $\boldsymbol{b}, \boldsymbol{c}$, and the matrix A of the LP (2.1) to be deterministically fixed. Similarly, we would not always expect the functions g_i's or the set $\mathscr{X}$ in the mathematical program (2.2) to be deterministically fixed.

For instance, the future productivities in a production problem, inflows into a reservoir connected to a hydro power station, and demands at various nodes in a transportation network, are often appropriately modelled as uncertainty parameters, which are best characterised by discrete or continuous probability distributions.

The uncertainty about the realised values of those parameters cannot always be ignoredby just inserting their mean values or some other (fixed) estimates during the modelling process. That is, depending on the practical

situation under consideration, problems like those specified by (2.1) may not be the appropriate models for describing what we want to solve. = We will demonstrate this using a basic idealised yet illustrative example of a refinery, which We will manipulate in various ways to show a variety of stochastic linear/non-linear models that you may encounter in the real world.

The goal is to essentially find a solution, which is feasible and in some sense, optimal. Such models are generally formulated, solved analytically or numerically, and analysed in order to provide useful information to a decision-maker.

***Example* 19** (Refinery)**.** A refinery produces STANDARD and PREMIUM from two raw crude oils, raw_1 and raw_2, which cost \$200 and \$300 per barrel respectively, giving the production cost for the products. The demand in gallons for (STANDARD, PREMIUM) per day is given by $\boldsymbol{h}^\top = (1800, 1620)$ and the refinery can process a maximum of 100 barrels of raw materials each day. The productivity (or output) of STANDARD is 20 gallons per barrel of raw_1 and 60 gallons per barrel of raw_2, whereas the productivity (output) of PREMIUM is 30 gallons per barrel of raw_1 and 30 gallons per barrel of raw_2.

We can write this down as a simple LP and solve for the minimum cost to the refinery to satisfy demand, by letting x_i be the number of barrels of raw_i, for $i = 1, 2$ processed by the refinery per day.

$$\left.\begin{array}{rrrrrrrrr} \min & w & = & 200x_1 & + & 300x_2 & & & \\ \text{(supply)} & \text{s.t.} & & x_1 & + & x_2 & \leq & 100 & \\ \text{(standard demand)} & & & 20x_1 & + & 60x_2 & \geq & 1800 & \\ \text{(premium demand)} & & & 30x_1 & + & 30x_2 & \geq & 1620 & \\ & & & x_1 \geq 0, & & x_2 \geq 0 & & & \end{array}\right\} \tag{2.3}$$

We could then solve this LP graphically or use the Simplex algorithm.

Graphical Solution:
As can be seen from Figure 2.1, the optimal solution is

$$\boldsymbol{x}^* = (x_1^*, x_2^*) = (36, 18),$$

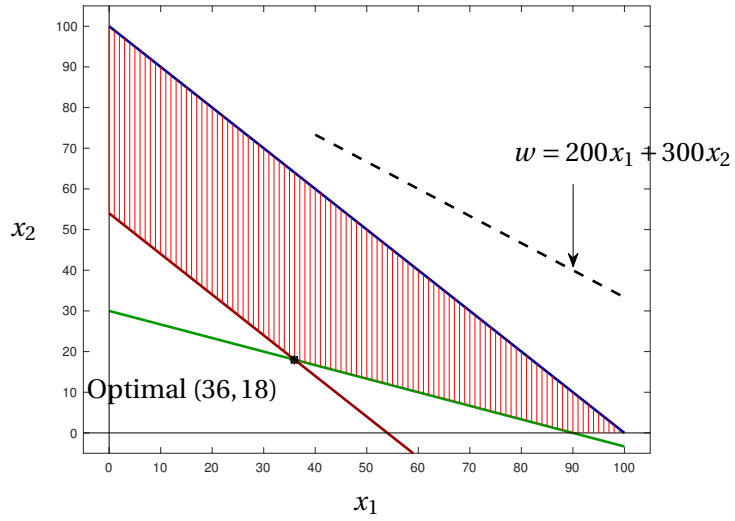


Figure 2.1: *Feasible solution space for Refinery LP , represented by the shaded redarea. The thick blue line corresponds to* $x_1 + x_2 = 100$*, the thick green line to* $20x_1 + 60x_2 = 1800$*, and the thick redline to* $30x_1 + 30x_2 = 1620$*. The slope of the objective function* $w = 200x_1 + 300x_2$ *is* $-2/3$*.*

with the optimal value $w^* = \$12,600$.

The **MATLAB solution** agrees with the graphical solution.

```
>> f=[200,300];
>> A=[1,1;-20,-60;-30,-30];
>> b=[100,-1800,-1620];
>> LB=[0;0];
>> [x,fval]=linprog(f,A,b,[],[],LB)
Optimization terminated.

x =
   36.0000
   18.0000
```

```
fval =
   1.2600e+04
```

Our production problem in Example 19 is properly described by a linear program, and its correct solution is as given above provided the productivities, the unit costs, the demands and the capacity are **fixed data** and **known to us prior to making our decision** on the production plan. However, these are obviously not always realistic assumptions. It may happen that at least some of the data, such as the productivities and demands vary within certain limits (for our discussion, randomly) and that we have to make our decision on the production plan before knowing the exact values of those data.

To be more specific, let us assume in the next example some specific variations that follow a known distributional form.

***Example* 20** (Refinery with Random Inputs)**.** Consider now the following, where some of the data varies randomly and we have to make our decision before we know the exact values:

- demand for STANDARD and PREMIUM varies randomly,
- productivities (outputs) of the fuels from raw products also vary randomly, and
- the outputs of fuel can only be observed during production.

At the same time, as with the case when we had all fixed and known data, the conditions are:

- daily production must be fixed in advance and cannot be changed,
- customers expect their actual demands to be met on a daily basis.

First, we assume that the demand in gallons for (STANDARD, PREMIUM) is now given by

$$\boldsymbol{h}^\top = \left(h_s, h_p\right) = (1800 + Z_1, 1620 + Z_2),$$

where $Z_1 \sim \mathcal{N}(0,120)$ and $Z_2 \sim \mathcal{N}(0,90)$. '$\mathcal{N}(\mu,\sigma)$' indicates a Normal distribution with mean μ and variance σ^2. In this case, the average demands remain equivalent to the deterministic demands of the original linear program; that is,

$$\mathbf{E}(1800 + Z_1) = 1800 \quad \text{and} \quad \mathbf{E}(1620 + Z_2) = 1620.$$

Let us also assume that the output of STANDARD from raw_1, α, and the output of PREMIUM from raw_2, β, are given by

$$\alpha = 20 + Y_1, \quad \beta = 34 - Y_2,$$

where $Y_1 \sim U[-8,8]$ and $Y_2 \sim exp(0.25)$. '$U[a,b]$' indicates uniform distribution on $[a,b]$, and 'exp(λ)' indicates exponential distribution with parameter λ. With these distributions, the average productivities remain equivalent to the deterministic productivities of the original linear program; that is,

$$\mathbf{E}(20 + Y_1) = 20 \quad \text{and} \quad \mathbf{E}(34 - Y_2) = 30.$$

Note that all random variables are assumed to be mutually independent. If we define $\boldsymbol{\zeta} := (Z_1, Z_2, Y_1, Y_2)^\top$ and let

$$\alpha(\boldsymbol{\zeta}) = 20 + Y_1, \;\; \beta(\boldsymbol{\zeta}) = 34 - Y_2, \;\; h_s(\boldsymbol{\zeta}) = 1800 + Z_1, \;\; h_p(\boldsymbol{\zeta}) = 1620 + Z_2,$$

we can write the 'stochastic' linear program

$$\left.\begin{array}{lrcrcrcl} \text{'min'} & w & = & 200x_1 & + & 300x_2 & & \\ \text{s.t.} & & & x_1 & + & x_2 & \leq & 100 \\ & & & \alpha(\boldsymbol{\zeta})x_1 & + & 60x_2 & \geq & h_s(\boldsymbol{\zeta}) \\ & & & 30x_1 & + & \beta(\boldsymbol{\zeta})x_2 & \geq & h_p(\boldsymbol{\zeta}) \\ & & & x_1 \geq 0, & & x_2 \geq 0. & & \end{array}\right\} \tag{2.4}$$

The 'stochastic' linear program given in (2.4) is not well defined, as the meaning of 'min' is not clear until we know the realisation $\boldsymbol{\varepsilon} = (z_1, z_2, y_1, y_2)^\top$ of the random vector $\boldsymbol{\zeta} = (Z_1, Z_2, Y_1, Y_2)^\top$. The random variables Z_1, Z_2 and Y_2

are also unbounded and so we restrict them to their 99% confidence intervals, to get the following 'allowable' realisations:

$$Z_1 \in [-309.1, 309.1], Z_2 \in [-231.8, 231.8], Y_1 \in [-8, 8], Y_2 \in [0.0, 18.42].$$

How should we deal with the randomness of Z_1, Z_2, Y_1, and Y_2?

Option 1. Suppose we fix $Y_1 = 0$ and $Y_2 = 4$ to their mean values, and use the 99% confidence interval limits of Z_1 and Z_2. The extreme values for each random variable yield

$$\begin{array}{lrcrcrcl}
\text{'min'} & w & = & 200x_1 & + & 300x_2 & & \\
\text{s.t.} & & & x_1 & + & x_2 & \leq & 100 \\
& & & 20x_1 & + & 60x_2 & \geq & 1800 - 309.1 \\
& & & 20x_1 & + & 60x_2 & \geq & 1800 + 309.1 \\
& & & 30x_1 & + & 30x_2 & \geq & 1620 - 231.8 \\
& & & 30x_1 & + & 30x_2 & \geq & 1620 + 231.8 \\
& & & x_1 \geq 0, & & x_2 \geq 0, & &
\end{array}$$

or, equivalently,

$$\begin{array}{lrcrcrcl}
\text{'min'} & w & = & 200x_1 & + & 300x_2 & & \\
\text{s.t.} & & & x_1 & + & x_2 & \leq & 100 \\
& & & 20x_1 & + & 60x_2 & \geq & 1490.9 \\
& & & 20x_1 & + & 60x_2 & \geq & 2109.1 \\
& & & 30x_1 & + & 30x_2 & \geq & 1388.2 \\
& & & 30x_1 & + & 30x_2 & \geq & 1851.8 \\
& & & x_1 \geq 0, & & x_2 \geq 0. & &
\end{array}$$

This results in a parallel shift in the averaged value facets of the original feasible region, shown in solid (redand green) lines, to the dotted lines in Figure 2.2 Note that the shaded redarea is feasible for all values of Z_1 and Z_2 (in their 99% confidence intervals).

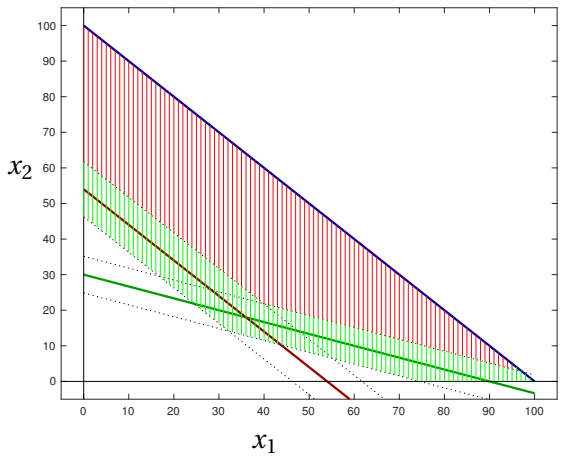


Figure 2.2: *Feasible solution space varying with demands*

Option 2: Alternatively, fix $Z_1 = 0$ and $Z_2 = 0$ to their mean values, and use the full distribution of Y_1 and the 99% confidence interval limits of Y_2. The extreme values for each random variable yield

$$
\begin{array}{llrcrcrcl}
\text{'min'} & w & = & 200x_1 & + & 300x_2 & & & \\
\text{s.t.} & & & x_1 & + & x_2 & \leq & 100 \\
& & & 12x_1 & + & 60x_2 & \geq & 1800 \\
& & & 28x_1 & + & 60x_2 & \geq & 1800 \\
& & & 30x_1 & + & 34x_2 & \geq & 1620 \\
& & & 30x_1 & + & 15.58x_2 & \geq & 1620 \\
& & & x_1 \geq 0, & & x_2 \geq 0. & & &
\end{array}
$$

This results in rotations (about the pivot points ⊗) of the facets of the original feasible region, shown in solid (redand green) lines, to the dotted lines in Figure 2.3. Note that the shaded red area is feasible for all values of Y_1 and Y_2.

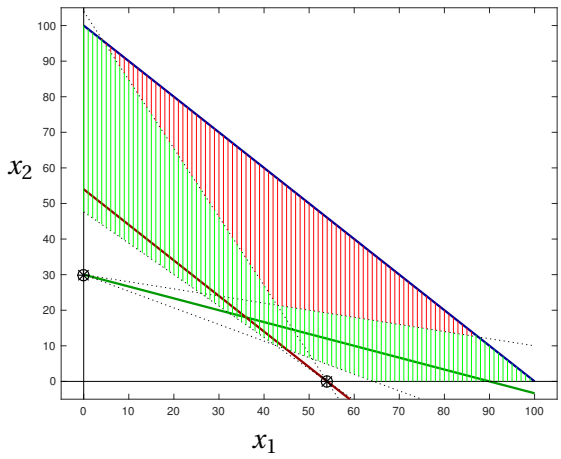


Figure 2.3: *Feasible solution space varying with productivities*

Option 3: Consider all possible changes in demands and productivities together, yielding

$$
\begin{array}{lrcrcrcl}
\text{`min'} & w & = & 200x_1 & + & 300x_2 & & \\
\text{s.t.} & & & x_1 & + & x_2 & \leq & 100 \\
& & & 12x_1 & + & 60x_2 & \geq & 1490.9 \\
& & & 12x_1 & + & 60x_2 & \geq & 2109.1 \\
& & & 28x_1 & + & 60x_2 & \geq & 1490.9 \\
& & & 28x_1 & + & 60x_2 & \geq & 2109.1 \\
& & & 30x_1 & + & 34x_2 & \geq & 1388.2 \\
& & & 30x_1 & + & 34x_2 & \geq & 1851.8 \\
& & & 30x_1 & + & 15.58x_2 & \geq & 1388.2 \\
& & & 30x_1 & + & 15.58x_2 & \geq & 1851.8 \\
& & & x_1 \geq 0, & & x_2 \geq 0. & &
\end{array}
$$

This results in both parallel translations and rotational movements, shown as dotted lines in Figure 2.4. The facets of the average value feasible region are shown in solid black.

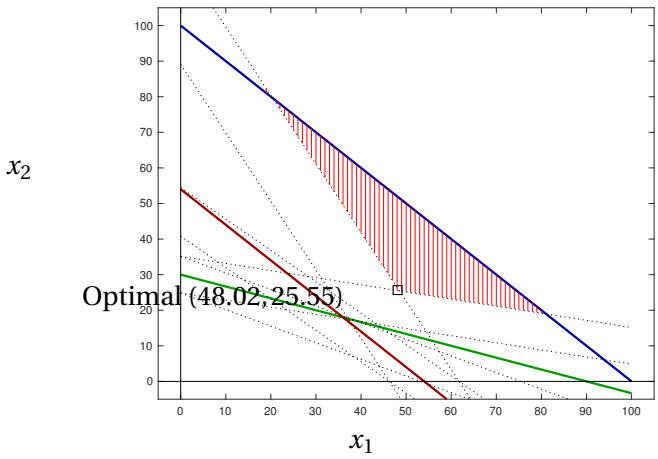


Figure 2.4: *Feasible solution space varying with productivities and demands*

Note that the feasible solution space depicted in Figure 2.4 is much contracted, being risk averse, and yields an **expensive optimal** solution of

$$\boldsymbol{x}^* = (x_1^*, x_2^*) = (48.02, 25.55)$$

with the corresponding optimal value $w^* = \$17,269.00$. This optimal solution is a 'fat' solution, which provides a 'safe' production program where the solution remains feasible under all possible realisations of the random variables.

The variation in the feasible region is substantial even for such a small problem, so what can we do?

§2.2 General Formulation

Defn 2.1. *In a **stochastic program**, we have that some of the coefficients are random variables. We might think of the problem as*

$$\begin{aligned} \text{`min'} \quad & z = g_0(\boldsymbol{x}, \boldsymbol{\zeta}) \\ \textit{such that} \quad & g_i(\boldsymbol{x}, \boldsymbol{\zeta}) \leq 0,\ i = 1, \ldots, m \\ & \boldsymbol{x} \in \mathscr{X} \subset \mathbb{R}^n \end{aligned} \tag{2.5}$$

where $\boldsymbol{\zeta}$ is a random vector. Note that $g_i(\boldsymbol{x}, \boldsymbol{\zeta})$ are random variables for all $\boldsymbol{x}$ and $i = 0, \ldots, m$.

Assumptions: The distribution function of $\boldsymbol{\zeta}$, $F_{\boldsymbol{\zeta}}(\cdot)$, is known, and $\boldsymbol{\zeta}$ is independent of the decision variables, $\boldsymbol{x}$.

The 'mathematical program' (2.5) is not well defined. The objective function $g_0(\boldsymbol{x}, \boldsymbol{\zeta})$ is a random variable, so what is the meaning of 'minimisation'? Given a realisation, $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$, we could then choose decision variables $\boldsymbol{x}$ to minimise $g_0(\boldsymbol{x}, \boldsymbol{\zeta})$ — a standard mathematical programming problem — but what about the situation where we wish, or need, to make a decision now, before the realisation $\boldsymbol{\zeta}$? Similarly, what is the interpretation of the constraints $g_i(\boldsymbol{x}, \boldsymbol{\zeta}) \leq 0$?

How might you handle such a problem?

One approach is to translate this problem into a *deterministic* problem, so the mathematical program is well defined. This leads to so-called **deterministic equivalent problems** (DEPs).

§2.3 NAÏVE DEP

Defn 2.2. *One DEP, which We will call the* ***naïve DEP****, is*

$$\begin{aligned} \min \quad & z = g_0(\boldsymbol{x}, \boldsymbol{\zeta}) \\ \textit{subject to} \quad & g_i(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq 0,\ i = 1, \ldots, m, \textit{ and for all realisations } \boldsymbol{\varepsilon} \textit{ of } \boldsymbol{\zeta}, \\ & \boldsymbol{x} \in \mathscr{X} \subset \mathbb{R}^n, \end{aligned} \tag{2.6}$$

where the objective function z is evaluated for the resultant $\boldsymbol{x}$ when every constraint under $\boldsymbol{\zeta}$ is satisfied.

Do you think this is sensible? Why, or why not?

§2.4 RECOURSE DEP

RECOURSE DEP: EXAMPLES

***Example* 21** (Refinery with Recourse)**.** Consider again our Refinery problem, where we now assume that the refinery has made the following arrangement with their customers.

The customers expect the refinery to satisfy their daily demands. However, due to the random nature of demands and productivity, the demands may not be covered. This will cause a 'penalty' cost on the refinery, as the shortfall must be purchased from the market. Assume that the penalty is proportional to the shortfall and that the price per gallon of the undeliverable fuels are as follows

$$\boldsymbol{q}^\top = \left(q_s, q_p\right) = (\$70, \$120).$$

These costs are due to shortage or in general due to a violation of the constraints and must be determined after the observation of the random data. They are referred to as **recourse costs**.

In the case of repeated execution of the production program (daily in our Refinery problem), it is meaningful to apply an **expected value criterion**.

That is, we may want to find a production plan that minimises the original **first stage costs** and the **expected recourse costs**.

For each of the two stochastic constraints in the model of equations (2.4), we introduce a recourse variable $y_i(\boldsymbol{\zeta})$ for $i = 1,2$, which measures the shortfall in production (it is like a slack variable), where recall that $\boldsymbol{\zeta} := (Z_1, Z_2, Y_1, Y_2)^\top$. Note that because the production shortfall depends on $\boldsymbol{\zeta}$, so do the corresponding recourse variables $y_i(\boldsymbol{\zeta})$. This makes the recourse variables themselves **random**.

We can now replace our vague stochastic program with a well defined stochastic program with recourse. That is, we include the recourse variables $y_1(\boldsymbol{\zeta})$ and $y_2(\boldsymbol{\zeta})$ in the inequalities and add the expected recourse cost w.r.t. $\boldsymbol{\zeta}$, given by

$$\mathbf{E}_{\boldsymbol{\zeta}}[70y_1(\boldsymbol{\zeta}) + 120y_2(\boldsymbol{\zeta})]$$

to the objective function to get the **two-stage linear recourse problem** given by:

$$\left.\begin{array}{llrcrcrcl} \text{`min'} & w & = & 200x_1 + 300x_2 + \mathbf{E}_{\boldsymbol{\zeta}}(70y_1(\boldsymbol{\zeta}) + 120y_2(\boldsymbol{\zeta})) & & & & & & \\ \text{s.t.} & & & x_1 & + & x_2 & & \leq & 100 \\ & & & \alpha(\boldsymbol{\zeta})x_1 & + & 60x_2 & + & y_1(\boldsymbol{\zeta}) & \geq & h_s(\boldsymbol{\zeta}) \\ & & & 30x_1 & + & \beta(\boldsymbol{\zeta})x_2 & + & y_2(\boldsymbol{\zeta}) & \geq & h_p(\boldsymbol{\zeta}) \\ & & & x_1 \geq 0, & & x_2 \geq 0, & & y_1(\boldsymbol{\zeta}) \geq 0, \; y_2(\boldsymbol{\zeta}) \geq 0 & & \end{array}\right\} \tag{2.7}$$

The problem given in (2.7) is an example of a deterministic equivalent problem known as a **Recourse DEP**. In general the DEP given in (2.7) must hold almost surely (a.s., or with probability 1).

***Example* 22** (Refinery with Recourse: Finite discrete variations)**.** If the random vector $\boldsymbol{\zeta}$ has a finite discrete distribution

$$\{(\boldsymbol{\varepsilon}_k, p_k), k = 1, \ldots, r\},$$

where $r < \infty$ and $p_k > 0$ for all k, we get a linear program written in **expanded form** as

$$\left.\begin{array}{rcl} \min w & = & 200x_1 + 300x_2 + \sum_{k=1}^{r} p_k \left[70y_1(\boldsymbol{\varepsilon}_k) + 120y_2(\boldsymbol{\varepsilon}_k)\right] \\ \text{s.t.} & & \begin{array}{rcrcrcl} x_1 & + & x_2 & & & \leq & 100 \\ \alpha(\boldsymbol{\varepsilon}_k)x_1 & + & 60x_2 & + & y_1(\boldsymbol{\varepsilon}_k) & \geq & h_s(\boldsymbol{\varepsilon}_k),\ \forall k \\ 30x_1 & + & \beta(\boldsymbol{\varepsilon}_k)x_2 & + & y_2(\boldsymbol{\varepsilon}_k) & \geq & h_p(\boldsymbol{\varepsilon}_k),\ \forall k \end{array} \\ & & x_1 \geq 0,\ x_2 \geq 0,\ y_1(\boldsymbol{\varepsilon}_k) \geq 0,\ y_2(\boldsymbol{\varepsilon}_k) \geq 0,\ \forall k. \end{array}\right\} \quad (2.8)$$

Because $\boldsymbol{\zeta}$ has a finite discrete distribution $\{(\boldsymbol{\varepsilon}_k, p_k), k = 1, \ldots, r\}$, where $p_k > 0$ for all k, we noted that the LP (2.8) is just an ordinary linear program, but it has what is called a **Dual Decomposition Structure**. Depending on the number r of realisations $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$, this linear program may become very large in scale (thus making it difficult or even impossible to solve in this form), but its particular block structure is amenable to specially designed algorithms. We will return to this later in the course.

Let's continue our example with continuous distributions $\boldsymbol{\zeta}$, where we will form discrete approximations.

***Example* 23** (Refinery with Recourse: Random Demands and Discretisation)**.** Initially, let's consider the situation where the demands $\boldsymbol{d}$ vary randomly as the normal distributions Z_1 and Z_2 and the productivities are fixed, similar to the situation as displayed in Figure 2.2. Even in this situation the non-linear LP (NLP) can present numerical difficulties as evaluation of the expectation in the objective function requires

- multivariate numerical integration,
- implicit definition of the functions $y_1(\boldsymbol{\varepsilon})$ and $y_2(\boldsymbol{\varepsilon})$ for every possible realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$.

To deal with these difficulties, We will approximate the normal distributions by finite discrete distributions, using a technique that can be applied to most continuous distributional forms.

In order to approximate the normal distributions we can

- generate samples z_1^μ and z_2^μ, from the random variables Z_1 and Z_2 respectively for $\mu = 1,2,\ldots,K$, restricted to the 99% confidence intervals (CIs) for some large value of K;
- create equi-spaced partitions of the 99% CIs into a finite number — say, r_i — of subintervals $I_{i,\nu}$ for $i = 1,2$;
- calculate for each subinterval the mean $\overline{z_{i,\nu}}$ of sample values $z_i^\mu \in I_{i,\nu}$ for $i = 1,2$ and $\nu = 1,2,\ldots r_i$, to use as an estimate of

$$\mathbf{E}(Z_i | Z_i \in I_{i,\nu});$$

- calculate for every subinterval $I_{i,\nu}$ the relative frequency $p_{i,\nu}$ for $z_i^\mu \in I_{i,\nu}$, which yields an estimate for

$$P(Z_i \in I_{i,\nu}).$$

The following figure displays a result of using $K = 100,000$ and $r_1 = r_2 = 15$ in this approximation method for the two normal distributions $\mathcal{N}(1800,120)$ and $\mathcal{N}(1620,90)$.

NOTE: We must always remain aware that using discrete approximations inevitably leads to some discretisation error.

With $K = 100,000$, $r_1 = r_2 = 15$, we get $15^2 = 225$ realisations for the joint distribution of $\boldsymbol{\zeta}$. Using these realisations in the expanded form LP (2.8), we obtain

$$\begin{aligned} \widetilde{w} &= 14084.3728 \\ \widetilde{\boldsymbol{x}} = (\widetilde{x}_1, \widetilde{x}_2) &= (38.5490, 20.5478) \\ &\quad \text{with first stage costs of} \\ 200\widetilde{x}_1 + 300\widetilde{x}_2 &= 13874.1504. \end{aligned}$$

The **empirical reliability** is defined to be the probability that $\widetilde{\boldsymbol{x}}$ is feasible such that the recourse variables are zero:

$$P(\alpha(\boldsymbol{\zeta})\widetilde{x}_1 + 60\widetilde{x}_2 \geq h_s(\boldsymbol{\zeta}),\ 30\widetilde{x}_1 + \beta(\boldsymbol{\zeta})\widetilde{x}_2 \geq h_p(\boldsymbol{\zeta})),$$

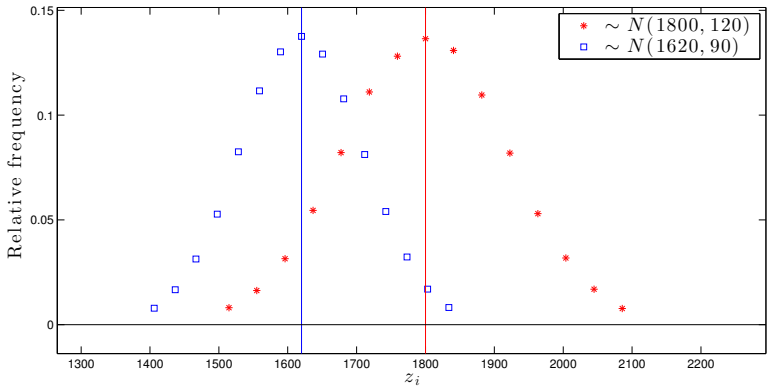


Figure 2.5: *Discrete approximation of demand distributions.*

where we use the discrete approximation of $\boldsymbol{\zeta}$.

In this case, **empirical reliability** is

$$\rho(\widetilde{\boldsymbol{x}}) = 0.9516.$$

If we force our original LP solution $\boldsymbol{x}^* = (36, 18)$ (that is, if we substitute the original LP solution into the expanded form LP), we see that

$$\begin{aligned} w^* &= 19980.0372 \\ &\text{with first stage costs of} \\ \boldsymbol{x}^*(200, 300)' &= 12600. \end{aligned}$$

The empirical reliability of this solution is $\rho(\boldsymbol{x}^*) = 0.3206$, which in this instance, far exceeds the theoretical expected reliability of 0.25 (half of the time each of the normal distributions are ≥ 0).

This indicates to some extent the discretisation errors that arise due to the approximation of the continuous distributions.

In the next example We will consider the same technique used when the productivities are random and the demands are fixed.

***Example* 24** (Refinery with Recourse: Random Productivities and Discretisation)**.** In the second instance, the demands are fixed and the productivities vary randomly as per the distributions of Y_1 and Y_2, similar to the situation as displayed in Figure 2.3. Similarly to before, we

- generate samples y_1^μ and y_2^μ, from the random variables Y_1 and Y_2 respectively for $\mu = 1,2,\ldots,K$, for some large value of K, where Y_2 is restricted to its 99% CI,
- create equi-spaced partitions of the data range intervals into a finite number—say, r_i—of subintervals $I_{i,v}$ for $i = 1,2$,
- calculate for each subinterval the mean $\overline{y_{i,v}}$ of sample values $y_i^\mu \in I_{i,v}$ for $i = 1,2$ and $v = 1,2,\ldots r_i$, to use as an estimate of

$$\mathbf{E}(Y_i | Y_i \in I_{i,v}),$$

- calculate for every subinterval $I_{i,v}$ the relative frequency $p_{i,v}$ for $y_i^\mu \in I_{i,v}$, which yields an estimate for

$$P(Y_i \in I_{i,v}).$$

The following figure displays a result of using $K = 100,000$, $r_1 = 15$ and $r_2 = 18$ in this approximation method for the two distributions $U(12,28)$ and $34 - exp(0.25)$.

Using these discrete distributions as approximations to the given uniform and exponential distributions, with $K = 100,000$, $r_1 = 15$ and $r_2 = 18$ we get $15 \times 18 = 270$ realisations for the joint distribution of $\boldsymbol{\zeta}$. Using this in the expanded form LP (2.8), yields

$$\begin{aligned} \widetilde{w} &= 14451.7938 \\ \widetilde{\boldsymbol{x}} = (x_1, x_2) &= (37.5610, 22.1520) \\ &\text{with first stage costs of} \\ 200\widetilde{x}_1 + 300\widetilde{x}_2 &= 14157.7893. \end{aligned}$$

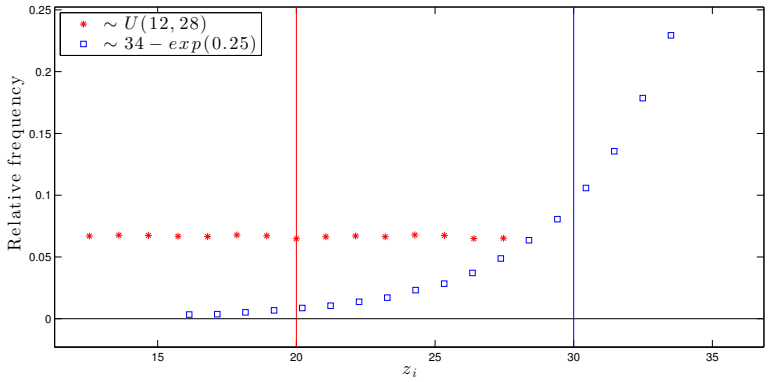


Figure 2.6: *Discrete approximation of productivities.*

The empirical reliability $\rho(\widetilde{\boldsymbol{x}}) = 0.9617$.

If we again force our original LP solution $\boldsymbol{x}^* = (36,18)$ we see that

$$\begin{aligned} w^* &= 20477.7807 \\ &\text{with first stage costs of} \\ \boldsymbol{x}^*(200,300)' &= 12600. \end{aligned}$$

The empirical reliability of solution is $\rho(\boldsymbol{x}^*) = 0.3020$.

***Example* 25** (Refinery with Recourse: Random Inputs and their Discrete Approximations)**.** Approximating all of the four random demand and productivity distributions, with $K = 100,000$, $r_1 = 5, r_2 = 9, r_3 = 7$ and $r_4 = 11$, we get $5 \times 9 \times 7 \times 11 = 3465$ realisations of the joint distribution of $\boldsymbol{\zeta}$.

Using these realisations in the expanded form LP (2.8), yields

$$\begin{aligned} \widetilde{w} &= 15058.2884 \\ \widetilde{\boldsymbol{x}} = (\widetilde{x}_1, \widetilde{x}_2) &= (37.8024, 23.6010) \\ &\text{with first stage costs of} \\ 200x_1 + 300x_2 &= 14640.7874. \end{aligned}$$

The empirical reliability is $\rho(\widetilde{\boldsymbol{x}}) = 0.9441$.

Again, if we force our original LP solution $\boldsymbol{x}^* = (36, 18)$ we see that

$$\begin{aligned} w^* &= 23183.6708 \\ &\text{with first stage costs of} \\ \boldsymbol{x}^*(200, 300)' &= 12600. \end{aligned}$$

The empirical reliability of this solution is $\rho(\boldsymbol{x}^*) = 0.2548$.

It is worth noting that for a relatively small increase in first stage costs, we have retrospectively calculated the empirical reliability for our recourse problems, which appear to have increased from 0.25 up to around 0.95.

Although these calculations may not provide a true indication of the size of the constraint violations, there are many situations where this form of reliability is a prime consideration, such as in medical and technical applications.

RECOURSE DEP: GENERAL FORMULATION

Defn 2.3. *For a given stochastic linear program, let us start by defining for $i = 1, \ldots, m$ and for each realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$ the following*

$$g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon}) := \begin{cases} 0 & \text{if } g_i(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq 0, \\ g_i(\boldsymbol{x}, \boldsymbol{\varepsilon}) & \text{otherwise.} \end{cases}$$

This implies the ith constraint of (2.5) is not satisfied if and only if $g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon}) > 0$ for a given decision $\boldsymbol{x}$ and realisation $\boldsymbol{\varepsilon}$. Furthermore, the extent of the violation in the ith constraint is $g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon})$.

If such violations occur for a given *a priori* choice of $\boldsymbol{x}$ and realisation $\boldsymbol{\varepsilon}$, it might be possible to undertake a **second-stage activity**, or **recourse**, in order to satisfy the constraint.

That is, we could provide for each constraint violation, a recourse or **second-stage** activity $y_i(\boldsymbol{\varepsilon})$ that, after observing the realisation $\boldsymbol{\varepsilon}$, is chosen such as to compensate for each constraint violation, if they exist, by satisfying

$$g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon}) - y_i(\boldsymbol{\varepsilon}) \leq 0,$$

or, equivalently,

$$y_i(\boldsymbol{\varepsilon}) \geq g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon}).$$

If this extra effort is assumed to cause an extra cost or penalty of q_i (≥ 0) per unit of violation, then our additional costs – that is, the **recourse function** – is

$$Q(\boldsymbol{x}, \boldsymbol{\varepsilon}) = \min_{\boldsymbol{y}} \left\{ \sum_{i=1}^{m} q_i y_i(\boldsymbol{\varepsilon}) : y_i(\boldsymbol{\varepsilon}) \geq g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon}), i = 1, \ldots, m \right\}, \tag{2.9}$$

(which is a linear program itself,) as we wish to satisfy the constraints with minimum expense. Hence, the *total cost* – composed of first stage costs plus the recourse costs – is

$$f_0(\boldsymbol{x}, \boldsymbol{\varepsilon}) = g_0(\boldsymbol{x}, \boldsymbol{\varepsilon}) + Q(\boldsymbol{x}, \boldsymbol{\varepsilon}). \tag{2.10}$$

Note that here we have one recourse variable per constraint.

Rather than (2.9), we might think of a more general linear recourse program with a different recourse vector

$$\boldsymbol{y} = (y_1, y_2, \ldots, y_{\overline{n}}) \in \mathscr{Y} \subset \mathbb{R}^{\overline{n}},$$

where $\mathscr{Y}$ is some given polytope (such as $\{\boldsymbol{y} | \boldsymbol{y} \geq \boldsymbol{0}\}$). We also have an arbitrary, fixed $m \times \overline{n}$ matrix W – the **recourse matrix** – and a corresponding unit cost vector $\boldsymbol{q} = (q_1, q_2, \ldots, q_{\overline{n}}) \in \mathbb{R}^{\overline{n}}$, yielding for (2.10) the recourse function:

$$Q(\boldsymbol{x}, \boldsymbol{\varepsilon}) = \min_{\boldsymbol{y}(\boldsymbol{\varepsilon})} \left\{ \boldsymbol{q}^\top \boldsymbol{y}(\boldsymbol{\varepsilon}) : W \boldsymbol{y}(\boldsymbol{\varepsilon}) \geq \boldsymbol{g}^+(\boldsymbol{x}, \boldsymbol{\varepsilon}), \boldsymbol{y}(\boldsymbol{\varepsilon}) \in \mathscr{Y} \right\}, \tag{2.11}$$

where $\boldsymbol{g}^+(\boldsymbol{x}, \boldsymbol{\varepsilon}) = (g_1^+(\boldsymbol{x}, \boldsymbol{\varepsilon}), \ldots, g_m^+(\boldsymbol{x}, \boldsymbol{\varepsilon}))^\top$ is the vector of constraint violations (recall that these may be 0).

The recourse vector here no longer directly coincides with one variable per constraint with an associated direct cost, but rather intends to satisfy the constraint violations using some other process.

The difference between the recourse functions (2.9) and (2.11) is in the mechanism by which we use recourse to satisfy any violations:

- The recourse function (2.9) could for instance be interpreted as buying the shortage of products at the market.
- Alternatively, the recourse function in (2.11) gives us the flexibility to model a *second-stage*, or an *emergency* production program, which is effected with *inputs* $\boldsymbol{y}$ and a *technology* (a procedure for converting inputs into products) represented by the matrix W.

Note that the recourse function (2.9) appears as a special case of (2.11); that is, the recourse function (2.9) is readily expressible in the form (2.11), with $W = I_{m \times m}$.

Finally, note that the recourse function could also be non-linear:

$$Q(\boldsymbol{x}, \boldsymbol{\varepsilon}) \quad = \quad \min_{\boldsymbol{y}} \left\{ q(\boldsymbol{y}) \;\middle|\; \begin{array}{c} H_i(\boldsymbol{y}) \geq g_i^+(\boldsymbol{x}, \boldsymbol{\varepsilon}), i = 1, \ldots, m \\ \boldsymbol{y} \in \mathscr{Y} \subset \mathbb{R}^{\overline{n}} \end{array} \right\}, \tag{2.12}$$

where $q(\boldsymbol{y}) : \mathbb{R}^{\overline{n}} \to \mathbb{R}$ and $H_i(\boldsymbol{y}) : \mathbb{R}^{\overline{n}} \to \mathbb{R}$ are assumed to be given.

In any case, if it is acceptable and meaningful to the decision maker to minimise the expected value of the total costs, instead of problem (2.5), we consider its deterministic equivalent, the **two-stage stochastic program with recourse**:

$$\min_{\boldsymbol{x}\in\mathscr{X}} \mathbf{E}_{\boldsymbol{\zeta}}(f_0(\boldsymbol{x},\boldsymbol{\zeta})) \quad = \quad \min_{\boldsymbol{x}\in\mathscr{X}} \mathbf{E}_{\boldsymbol{\zeta}}(g_0(\boldsymbol{x},\boldsymbol{\zeta}) + Q(\boldsymbol{x},\boldsymbol{\zeta})) \tag{2.13}$$

where $\mathbf{E}_{\boldsymbol{\zeta}}(\cdot)$ means the expectation with respect to $\boldsymbol{\zeta}$.

RECOURSE DEP: MULTI-STAGE RECOURSE PROBLEM

This approach can be extended to a **multi-stage recourse problem**. That is, instead of two decisions $\boldsymbol{x}$ and $\boldsymbol{y}$, to be taken at stages 1 and 2, we would have $K+1$ sequential decisions $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_K$, where $\boldsymbol{x}_\tau \in \mathbb{R}^{\overline{n}_\tau}$ for $\tau = 1, \ldots, K$, to be taken at the subsequent stages $\tau = 0, 1, \ldots, K$.

Let's assume that the objective function from (2.5) is deterministic, so that $g_0(\boldsymbol{x},\boldsymbol{\zeta}) \equiv g_0(\boldsymbol{x})$. At stage $\tau \geq 1$, we know the realisations $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_\tau$ of the random vectors $\boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_\tau$, as well as the previous decision vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\tau-1}$, and we have to decide on $\boldsymbol{x}_\tau$ such that the constraint(s) with vector-valued constraint functions

$$\boldsymbol{g}_K(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\tau, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_\tau)$$

are satisfied. At this stage, these can only be achieved by the proper choice of $\boldsymbol{x}_\tau$, based on the knowledge of the previous decisions and realizations.

Hence, given a cost function $\boldsymbol{q}_\tau(\boldsymbol{x}_\tau)$ at stage $\tau \geq 1$, we have a recourse function

$$\begin{aligned} &Q_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\tau-1}, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_\tau) \\ &= \min_{\boldsymbol{x}_\tau} \left\{ \boldsymbol{q}_\tau(\boldsymbol{x}_\tau) \middle| \boldsymbol{g}_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\tau, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_\tau) \leq 0 \right\}, \end{aligned} \tag{2.14}$$

where $\boldsymbol{g}_\tau(\cdot)$ is an m-vector of constraints at stage τ such that the term "stages" can, but need not, be interpreted as "time periods".

The recourse function (2.14) indicates that the optimal recourse action $\widehat{\boldsymbol{x}}_\tau$ at time τ depends on the previous decisions and on the realizations observed until stage τ. That is,

$$\widehat{\boldsymbol{x}}_\tau = \widehat{\boldsymbol{x}}_\tau \left(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\tau-1}, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_\tau\right), \qquad \tau \geq 1.$$

Hence, taking into account the multiple stages, we get the total costs for the multistage problem as

$$f_0 \left(\boldsymbol{x}_0, \boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2, \ldots, \boldsymbol{\varepsilon}_K\right) = g_0(\boldsymbol{x}_0) + \sum_{\tau=1}^{K} Q_\tau \left(\boldsymbol{x}_0, \widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_{\tau-1}, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_\tau\right).$$

This yields the deterministic equivalent for the described dynamic decision problem, **the multistage stochastic program with recourse**, as

$$\min_{\boldsymbol{x}_0 \in \mathscr{X}} \mathbf{E}_{\boldsymbol{\zeta}} \left(g_0(\boldsymbol{x}_0) + \sum_{\tau=1}^{K} Q_\tau \left(\boldsymbol{x}_0, \widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_{\tau-1}, \boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_\tau\right) \right),$$

which is a straight generalisation of the former (two-stage) stochastic program with recourse given in (2.13).

STOCHASTIC LINEAR PROGRAMS WITH RECOURSE

Let us now focus on *stochastic linear programs with recourse*. Assume that we are given the following stochastic linear program

$$\begin{aligned} \text{minimise} \quad & w = \boldsymbol{c}^\top \boldsymbol{x} \\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & T(\boldsymbol{\zeta})\boldsymbol{x} = \boldsymbol{h}(\boldsymbol{\zeta}), \\ & \boldsymbol{x} \geq \boldsymbol{0}. \end{aligned} \tag{2.15}$$

The $m_1 \times n$ matrix $T(\boldsymbol{\zeta})$ and the vector $h(\boldsymbol{\zeta})$ can depend on vector $\boldsymbol{\zeta}$ and hence can have random entries. The $m_0 \times n$ matrix A and the vector $\boldsymbol{b}$, on the other hand, are deterministic.

In general, we will assume that the dependence on $\boldsymbol{\zeta} \in \mathscr{S}_{\boldsymbol{\zeta}} \subset \mathbb{R}^k$ is of the form

$$\begin{aligned} T(\boldsymbol{\zeta}) &= \widehat{T}_0 + \sum_{i=1}^{k} 1_{\{\boldsymbol{\varepsilon}_i\}} \widehat{T}_i, \\ \boldsymbol{h}(\boldsymbol{\zeta}) &= \widehat{\boldsymbol{h}}_0 + \sum_{i=1}^{k} 1_{\{\boldsymbol{\varepsilon}_i\}} \widehat{\boldsymbol{h}}_i, \end{aligned} \tag{2.16}$$

with deterministic matrices $\widehat{T}_0, \widehat{T}_1, \ldots, \widehat{T}_k$ and deterministic vectors $\widehat{\boldsymbol{h}}_0, \widehat{\boldsymbol{h}}_1, \ldots, \widehat{\boldsymbol{h}}_k$, where $1_{\{\boldsymbol{\varepsilon}_i\}}$ is the indicator function that realisation $\boldsymbol{\varepsilon}_i$ happens. We now have a **stochastic linear program with fixed recourse** defined as follows.

Defn 2.4. *A **two-stage stochastic linear program with fixed recourse** has the following form:*

$$\begin{aligned} \min \quad & w = \boldsymbol{c}^\top \boldsymbol{x} + \mathbf{E}_{\boldsymbol{\zeta}}[Q(\boldsymbol{x},\boldsymbol{\zeta})] \\ \textit{such that} \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{x} \geq \mathbf{0}, \end{aligned} \tag{2.17}$$

where for each realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$,

$$Q(\boldsymbol{x},\boldsymbol{\varepsilon}) = \min_{\boldsymbol{y}} \left\{ \boldsymbol{q}^\top \boldsymbol{y} : W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x},\ \boldsymbol{y} \geq \mathbf{0} \right\}. \tag{2.18}$$

Note that for each realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$, $Q(\boldsymbol{x},\boldsymbol{\varepsilon})$ is just a linear program

$$\begin{aligned} \min \quad & \boldsymbol{q}^\top \boldsymbol{y} \\ \text{subject to} \quad & W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x}, \\ & \boldsymbol{y} \geq \mathbf{0}. \end{aligned}$$

Recall that if $\boldsymbol{\zeta}$ has a finite discrete distribution $\{(\boldsymbol{\varepsilon}_k, p_k), k = 1,\ldots,r\}$, where $p_k > 0$ for all k, we can write it out in expanded form as follows.

Defn 2.5. *The **expanded deterministic equivalent problem form** is:*

$$\left.\begin{aligned} \min \quad w &= \boldsymbol{c}^\top \boldsymbol{x} + \sum_{k=1}^{r} p_k \boldsymbol{q}^\top \boldsymbol{y}_k \\ \textit{subject to} \quad A\boldsymbol{x} &= \boldsymbol{b} \\ T(\boldsymbol{\varepsilon}_k)\boldsymbol{x} + W\boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{\varepsilon}_k) \quad \textit{for } k = 1,\ldots,r \\ \boldsymbol{x}, \boldsymbol{y}_k &\geq \mathbf{0}, \quad \textit{for } k = 1,\ldots,r. \end{aligned}\right\} \tag{2.19}$$

Note that the above deterministic equivalent problem (DEP) is just a simple LP, which if small enough can be directly solved.

Defn 2.6. *In particular, we have so-called* ***complete fixed recourse*** *if the fixed $m_1 \times \overline{n}$ recourse matrix W satisfies*

$$\{\boldsymbol{z} : \boldsymbol{z} = W\boldsymbol{y},\ \boldsymbol{y} \geq \boldsymbol{0}\} = \mathbb{R}^{m_1},$$

where m_1 is the number of stochastic constraints, and $\overline{n}$ is the number of recourse variables (that is, the dimension of the recourse vector $\boldsymbol{y}$).

This implies that, whatever the first-stage decision $\boldsymbol{x}$ and realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$, the second-stage program

$$Q(\boldsymbol{x}, \boldsymbol{\varepsilon}) = \min_{\boldsymbol{y}} \{\boldsymbol{q}^\top \boldsymbol{y} : W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x},\ \boldsymbol{y} \geq \boldsymbol{0}\}$$

will always be feasible.

Defn 2.7. *A special case of complete fixed recourse is called* ***simple recourse*** *where*

$$W = (I, -I)$$

with I the identity matrix of order m_1, and we have two non-negative second stage variables $y^+, y^- \geq 0$ for each of the m_1 first stage constraints.

In simple recourse, we also take account of the oversupply of commodities in the second stage by applying a penalty for such an occurrence as well as for the subsequent supply of the shortfall.

Here, the second stage reads as

$$Q(\boldsymbol{x}, \boldsymbol{\varepsilon}) = \begin{cases} \min & (\boldsymbol{q}^+)^\top \boldsymbol{y}^+ + (\boldsymbol{q}^-)^\top \boldsymbol{y}^- \\ \text{s.t.} & \boldsymbol{y}^+ - \boldsymbol{y}^- = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x}, \\ & \boldsymbol{y}^+,\ \boldsymbol{y}^- \geq \boldsymbol{0}, \end{cases}$$

where the unit cost of shortage for commodity i is given by q_i^+ and the unit cost of oversupply for product j is given by q_j^-.

Then for $\boldsymbol{q}^+ + \boldsymbol{q}^- \geq \mathbf{0}$ the recourse variables $\boldsymbol{y}^+$ and $\boldsymbol{y}^-$ can be chosen to measure (positively) the absolute deficiencies in the stochastic constraints. That is, the absolute measure of undersupply and oversupply.

***Example* 26** (Simple recourse)**.** Consider

$$\begin{array}{rl} \min & w = -x_2 \\ \text{subject to} & x_1 + x_2 + x_3 = 2 \\ & a_1 x_1 + a_2 x_2 + x_4 = 2 \\ & -1 \leq x_1 \leq 1, x_i \geq 0, \text{ for } i = 2,3,4, \end{array}$$

where

$$(a_1, a_2) = \begin{cases} (1, 3/4) & \text{with probability } 1/2, \\ (-3, 5/4) & \text{with probability } 1/2. \end{cases}$$

Note that $\mathbf{E}(a_1) = -1$, $\mathbf{E}(a_2) = 1$, and the deterministic LP with $a_1 = -1$ and $a_2 = 1$ has solution $w^* = -2$, with $\boldsymbol{x}^* = (0,2,0,0)^\top$.

Suppose that we have a recourse policy that allows compensation (incurs a penalty) of 5 per unit violation of the constraint

$$a_1 x_1 + a_2 x_2 + x_4 = 2$$

from its value 2. This penalty can be expressed as

$$5\,|2 - (a_1 x_1 + a_2 x_2 + x_4)|\,.$$

Thus, we can restate our problem in the following way

$$\begin{array}{rl} \min & w = -x_2 + 5\mathbf{E}\,(|2 - (a_1 x_1 + a_2 x_2 + x_4)|) \\ \text{subject to} & x_1 + x_2 + x_3 = 2 \\ & -1 \leq x_1 \leq 1, x_i \geq 0 \text{ for } i = 2,3,4, \end{array}$$

and the expectation may be written as

$$\frac{1}{2}(y_1^+ + y_1^-) + \frac{1}{2}(y_2^+ + y_2^-), \quad \text{where } y_j^+, y_j^- \geq 0 \text{ for } j = 1,2.$$

Hence, in expanded form we may write

$$\begin{aligned} \min \quad & w = -x_2 + \frac{5}{2}\left(y_1^+ + y_1^- + y_2^+ + y_2^-\right) \\ \text{subject to} \quad & x_1 + x_2 + x_3 = 2 \\ & x_1 + 3/4x_2 + x_4 + y_1^+ - y_1^- = 2 \\ & -3x_1 + 5/4x_2 + x_4 + y_2^+ - y_2^- = 2 \\ -1 \le x_1 \le 1, \quad & x_i,\ y_j^+, y_j^- \ge 0, \ \text{for } i = 2,3,4, \text{ and } j = 1,2, \end{aligned}$$

which has the optimal solution $\boldsymbol{x}^* = (0.2222, 1.7778, 0.0000, 0.4444)^\top$ and the corresponding optimal value $w^* = -1.7778$.

As a two stage DEP, we may write

$$\begin{aligned} \min \quad & w = -x_2 + 5\mathbf{E}(Q(\boldsymbol{x}, \boldsymbol{\zeta})) \\ \text{s.t.} \quad & x_1 + x_2 + x_3 = 2 \\ -1 \le x_1 \le 1, & \, x_i \ge 0 \text{ for } i = 2,3,4, \end{aligned}$$

where

$$\mathbf{E}(Q(\boldsymbol{x}, \boldsymbol{\zeta})) = 1/2\big(Q(\boldsymbol{x}, \boldsymbol{\varepsilon}_1) + Q(\boldsymbol{x}, \boldsymbol{\varepsilon}_2)\big),$$

such that

$$\begin{aligned} Q(\boldsymbol{x}, \boldsymbol{\varepsilon}_1) \quad : \quad \min \quad & y_1^+ + y_1^- \\ \text{s.t.} \quad & y_1^+ - y_1^- = 2 - \left(x_1 + 3/4x_2 + x_4\right) \\ & y_1^+, y_1^- \ge 0 \qquad \text{and} \\ Q(\boldsymbol{x}, \boldsymbol{\varepsilon}_2) \quad : \quad \min \quad & y_2^+ + y_2^- \\ \text{s.t.} \quad & y_2^+ - y_2^- = 2 - \left(-3x_1 + 5/4x_2 + x_4\right) \\ & y_2^+, y_2^- \ge 0. \end{aligned}$$

RECOURSE DEP: PROPERTIES

A natural question which then arises is: what properties do these problems have, and in particular, are they of a class which can be solved? Some important properties to investigate further are *convexity*, and *smoothness*, such that

we have some chance to deal with them, and confidence in the results, using the toolkit of mathematical programming methods. We will now consider such issues.

Convexity for the recourse problem (2.6) can be shown under rather mild conditions, assuming the integrability of $g_0 + Q$.

Proposition 2.1. *If $g_0(\boldsymbol{x},\boldsymbol{\varepsilon})$ and $Q(\boldsymbol{x},\boldsymbol{\varepsilon})$ are convex in $\boldsymbol{x}$ for each $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$, where $\mathscr{S}_{\boldsymbol{\zeta}}$ is the support of $\boldsymbol{\zeta}$, and if $\mathscr{X}$ is a convex set, then* (2.13),

$$\min_{\boldsymbol{x}\in\mathscr{X}} \mathbf{E}_{\boldsymbol{\zeta}}(f_0(\boldsymbol{x},\boldsymbol{\zeta})) = \min_{\boldsymbol{x}\in\mathscr{X}} \mathbf{E}_{\boldsymbol{\zeta}}(g_0(\boldsymbol{x},\boldsymbol{\zeta}) + Q(\boldsymbol{x},\boldsymbol{\zeta}))$$

is a convex program.

Proof. For $\widehat{\boldsymbol{x}},\overline{\boldsymbol{x}} \in \mathscr{X}$ and $\lambda \in (0,1)$, let $\widetilde{\boldsymbol{x}} := \lambda\widehat{\boldsymbol{x}} + (1-\lambda)\overline{\boldsymbol{x}}$. We have

$$g_0(\widetilde{\boldsymbol{x}},\boldsymbol{\varepsilon}) + Q(\widetilde{\boldsymbol{x}},\boldsymbol{\varepsilon}) \leq \lambda\left[g_0(\widehat{\boldsymbol{x}},\boldsymbol{\varepsilon}) + Q(\widehat{\boldsymbol{x}},\boldsymbol{\varepsilon})\right] + (1-\lambda)\left[g_0(\overline{\boldsymbol{x}},\boldsymbol{\varepsilon}) + Q(\overline{\boldsymbol{x}},\boldsymbol{\varepsilon})\right],$$

for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$, since $g_0(\boldsymbol{x},\boldsymbol{\varepsilon})$ and $Q(\boldsymbol{x},\boldsymbol{\varepsilon})$ are convex in $\boldsymbol{x}$ for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$. This implies, by taking expectations, that

$$\begin{aligned}\mathbf{E}_{\boldsymbol{\zeta}}\left[g_0(\widetilde{\boldsymbol{x}},\boldsymbol{\zeta}) + Q(\widetilde{\boldsymbol{x}},\boldsymbol{\zeta})\right] \leq \\ \lambda\mathbf{E}_{\boldsymbol{\zeta}}\left[g_0(\widehat{\boldsymbol{x}},\boldsymbol{\zeta}) + Q(\widehat{\boldsymbol{x}},\boldsymbol{\zeta})\right] + (1-\lambda)\mathbf{E}_{\boldsymbol{\zeta}}\left[g_0(\overline{\boldsymbol{x}},\boldsymbol{\zeta}) + Q(\overline{\boldsymbol{x}},\boldsymbol{\zeta})\right].\end{aligned}$$

□

Remark. For $\mathscr{Y} = \mathbb{R}^{\overline{n}}_+$ in the linear case (2.17) the convexity of $Q(\boldsymbol{x},\boldsymbol{\varepsilon})$ can immediately be asserted. For $\mathscr{Y} = \mathbb{R}^{\overline{n}}_+$ in the nonlinear case (2.12) the convexity of $Q(\boldsymbol{x},\boldsymbol{\varepsilon})$ holds if the functions $q(\boldsymbol{y})$ and $g_i(\boldsymbol{x},\boldsymbol{\varepsilon})$ are convex and the functions $H_i(\boldsymbol{y})$ are concave.

A sketch of the latter argument is as follows. Assume that $\widehat{\boldsymbol{y}}$ and $\overline{\boldsymbol{y}}$ solve (2.12) for $\widehat{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}$ respectively, at some realisation $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$. (That is, $\widehat{\boldsymbol{y}}$ and $\overline{\boldsymbol{y}}$

are optimal solutions of (2.12) for $\widehat{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}$ respectively, at some realisation $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$.)

Then, by the convexity of g_i and concavity of H_i, we have, for any $\lambda \in (0,1)$,

$$\begin{aligned} g_i(\lambda\widehat{\boldsymbol{x}} + (1-\lambda)\overline{\boldsymbol{x}}, \boldsymbol{\varepsilon}) &\leq \lambda g_i(\widehat{\boldsymbol{x}}, \boldsymbol{\varepsilon}) + (1-\lambda) g_i(\overline{\boldsymbol{x}}, \boldsymbol{\varepsilon}) \\ &\leq \lambda H_i(\widehat{\boldsymbol{y}}) + (1-\lambda) H_i(\overline{\boldsymbol{y}}) \\ &\leq H_i(\lambda\widehat{\boldsymbol{y}} + (1-\lambda)\overline{\boldsymbol{y}}). \end{aligned}$$

Hence, $\widetilde{\boldsymbol{y}} = \lambda\widehat{\boldsymbol{y}} + (1-\lambda)\widetilde{\boldsymbol{y}}$ is feasible in (2.12) for $\widetilde{\boldsymbol{x}} = \lambda\widehat{\boldsymbol{x}} + (1-\lambda)\overline{\boldsymbol{x}}$, and therefore by the convexity of q,

$$\begin{aligned} Q(\widetilde{\boldsymbol{x}}, \boldsymbol{\varepsilon}) &\leq q(\widetilde{\boldsymbol{y}}) \\ &\leq \lambda q(\widehat{\boldsymbol{y}}) + (1-\lambda) q(\widetilde{\boldsymbol{y}}) \\ &= \lambda Q(\widehat{\boldsymbol{x}}, \boldsymbol{\varepsilon}) + (1-\lambda) Q(\widetilde{\boldsymbol{x}}, \boldsymbol{\varepsilon}). \end{aligned}$$

□

Smoothness (i.e. partial differentiability) of recourse problems may also be asserted under fairly general conditions. However, We will not consider this property in this course. See [**?**] for further details.

Summary. In summary, stochastic programs with recourse often have the properties of convexity, and given continuous distributions differentiability, both highly beneficial from a mathematical programming perspective.

RECOURSE DEP: INDUCED CONSTRAINTS

Let $supp(P)$ denote the support of the probability measure P, i.e. the smallest closed set $\mathscr{S}$ such that $P(\mathscr{S}) = 1$. With the practical interpretation given to recourse problems in earlier lectures, and assuming $\mathscr{S}_{\boldsymbol{\zeta}} = supp(P_{\boldsymbol{\zeta}})$, we should hope that for any first-stage decision $\boldsymbol{x} \in \mathscr{X}$ there exists recourse variables $\boldsymbol{y} \in \mathscr{Y}$ for all realisations $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$. In other words, we hope that the

program

$$\begin{aligned} \text{minimise} \quad & Q(\boldsymbol{x},\boldsymbol{\varepsilon}) = \boldsymbol{q}^\top \boldsymbol{y} \\ \text{such that} \quad & W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x}, \\ & \boldsymbol{y} \geq \boldsymbol{0} \end{aligned} \tag{2.20}$$

is feasible $\forall \boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$.

However, depending upon the defined recourse matrix W and the given support $\mathscr{S}_{\boldsymbol{\zeta}}$, there might not exist a feasible solution $\boldsymbol{y} \in \mathscr{Y}$ for all first-stage decisions $\boldsymbol{x} \in \mathscr{X}$. Hence, it may be necessary to impose — in addition to $\boldsymbol{x} \in \mathscr{X}$ — further restrictions on our first-stage decisions, $\boldsymbol{x}$, called **induced constraints**.

Let us assume that $\mathscr{S}_{\boldsymbol{\zeta}}$ is a convex polytope, i.e. the **convex hull of finitely many points** $\boldsymbol{\varepsilon}_j \in \mathscr{S}_{\boldsymbol{\zeta}} \subset \mathbb{R}^k$, $j = 1,\ldots,r$:

$$\begin{aligned} \mathscr{S}_{\boldsymbol{\zeta}} &= \operatorname{conv}\{\boldsymbol{\varepsilon}_1,\ldots,\boldsymbol{\varepsilon}_r\} \\ &= \left\{ \boldsymbol{\varepsilon} : \boldsymbol{\varepsilon} = \sum_{j=1}^{r} \lambda_j \boldsymbol{\varepsilon}_j,\ \sum_{j=1}^{r} \lambda_j = 1,\ \lambda_j \in [0,1] \text{ for all } j = 1,\ldots,r \right\}. \end{aligned}$$

It follows that $\boldsymbol{x} \in \mathbb{R}^n$ allows for a feasible solution of the second stage program for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$ if and only if this is true for all $\boldsymbol{\varepsilon}_j$ for $j = 1,\ldots,r$. In other words, the corresponding induced first-stage feasibility set $\mathscr{K}$ is given as

$$\mathscr{K} = \left\{ \boldsymbol{x} : T(\boldsymbol{\varepsilon}_j)\boldsymbol{x} + W\boldsymbol{y}_j = \boldsymbol{h}(\boldsymbol{\varepsilon}_j),\ \boldsymbol{y}_j \geq \boldsymbol{0},\ j = 1,\ldots,r \right\}.$$

Theorem 2.1. *If the support $\mathscr{S}_{\boldsymbol{\zeta}}$ of $\boldsymbol{\zeta}$ is either a finite set or a convex polytope, then the **induced first-stage feasibility set** $\mathscr{K}$ is a convex polytope.*

The first-stage decision variables are then restricted to

$$\boldsymbol{x} \in \mathscr{X} \cap \mathscr{K}.$$

***Example* 27.** Consider the following first-stage feasibility set

$$\mathscr{X} = \{\boldsymbol{x} \in \mathbb{R}^2_+ : x_1 - 2x_2 \geq -4,\ x_1 + 2x_2 \leq 8,\ 2x_1 - x_2 \leq 6\}.$$

For the second-stage constraints, we have

$$W = \begin{pmatrix} -1 & 3 & 5 \\ 2 & 2 & 2 \end{pmatrix}, \quad T(\boldsymbol{\varepsilon}) = T = \begin{pmatrix} 2 & 3 \\ 3 & 1 \end{pmatrix}, \quad \boldsymbol{h}(\boldsymbol{\varepsilon}) = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \end{pmatrix},$$

and assume $\boldsymbol{\zeta}$ has support $\mathscr{S}_{\boldsymbol{\zeta}} = [4,19] \times [13,21]$. Then, the constraints to be satisfied for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$ are

$$W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T\boldsymbol{x},\ \boldsymbol{y} \geq \boldsymbol{0}. \tag{2.21}$$

Note that the second column $W_{:,2}$ of W is a linear combination of columns 1 and 3, that is,

$$W_{:,2} = \frac{1}{3}W_{:,1} + \frac{2}{3}W_{:,3},$$

and so (2.21) can be reduced to

$$\boldsymbol{h}(\boldsymbol{\varepsilon}) - T\boldsymbol{x} = \lambda W_{:,1} + \mu W_{:,3} \quad \text{for } \lambda, \mu \geq 0.$$

That is,

$$\begin{aligned} \varepsilon_1 - 2x_1 - 3x_2 &= -\lambda + 5\mu, \\ \varepsilon_2 - 3x_1 - x_2 &= 2\lambda + 2\mu, \\ \lambda, \mu \geq 0, &\quad \text{for all } \boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}. \end{aligned} \tag{2.22}$$

Now, by multiplying the constraints using the transformation matrix

$$S = \begin{pmatrix} 2 & 1 \\ -2 & 5 \end{pmatrix},$$

we obtain

$$S\boldsymbol{h}(\boldsymbol{\varepsilon}) - ST\boldsymbol{x} = S\begin{pmatrix} -\lambda & 5\mu \\ 2\lambda & 2\mu \end{pmatrix},$$

corresponding to adding two times the first equation to the second and -2 the first equations to 5 times the second, respectively. Then, we arrive at the *equivalent* system

$$\begin{aligned} 2\varepsilon_1 + \varepsilon_2 - 7x_1 - 7x_2 &= 12\mu, \\ -2\varepsilon_1 + 5\varepsilon_2 - 11x_1 + x_2 &= 12\lambda, \\ \lambda, \mu \geq 0, \forall \boldsymbol{\varepsilon} &\in \mathscr{S}_{\boldsymbol{\zeta}}. \end{aligned} \tag{2.23}$$

Furthermore, by noting that (2.23) has to hold for some $\lambda, \mu \geq 0$, we can rearrange this system of inequalities into an equivalent system for which we can propose some solutions. In particular, let $\lambda = \mu = 0$, we obtain

$$\begin{aligned} 7x_1 + 7x_2 &\leq 2\varepsilon_1 + \varepsilon_2, \\ 11x_1 - x_2 &\leq -2\varepsilon_1 + 5\varepsilon_2,, \end{aligned}$$

for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$. Then, as these inequalities have to hold for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$, we can choose the minimal right-hand sides to give the induced constraints

$$\mathscr{K} = \{\boldsymbol{x} : 7x_1 + 7x_2 \leq 21,\ 11x_1 - x_2 \leq 27\}.$$

Graphically, the first-stage feasibility set $\mathscr{X}$ together with the induced feasible set $\mathscr{K}$ is shown in Figure 2.7. The restricted feasibility set for the problem is then $\mathscr{X} \cap \mathscr{K}$.

Finally, for *complete fixed recourse*, for which $\mathscr{K} = \mathbb{R}^n$, the problem of induced constraints does not exist, since $(\mathscr{X} \cap \mathscr{K}) = \mathscr{X}$. This leads to the question: when do we have a stochastic program with complete fixed recourse? The following proposition addresses this question.

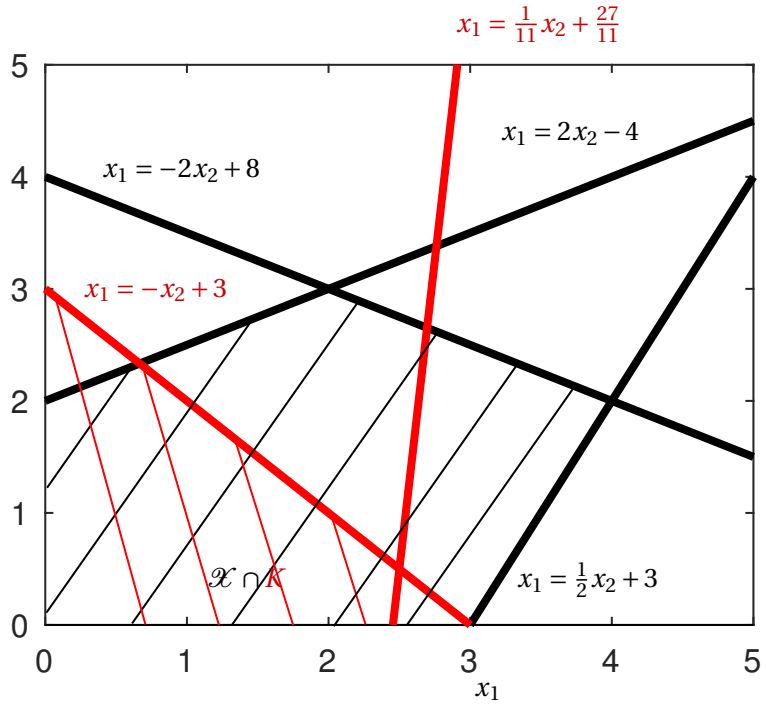


Figure 2.7: *The first-stage feasibility set $\mathcal{X}$ together with the induced feasible set $\mathcal{K}$.*

Proposition 2.2. *An $m_1 \times \overline{n}$ matrix W is a complete recourse matrix if and only if*

1. *it has rank$(W) = m_1$, and*
2. *assuming without loss of generality that its first m_1 columns $W_{:,1}, \ldots, W_{:,m_1}$ are linearly independent, the linear constraints*

$$\begin{aligned} W\boldsymbol{y} &= \mathbf{0} \\ y_i &\geq 1 \quad \text{for } i = 1, \ldots, m_1 \\ \boldsymbol{y} &\geq \mathbf{0} \end{aligned} \tag{2.24}$$

have a feasible solution.

Proof. W is a complete recourse matrix if and only if

$$\{\boldsymbol{z} : \boldsymbol{z} = W\boldsymbol{y},\ \boldsymbol{y} \geq \mathbf{0}\} = \mathbb{R}^{m_1}.$$

From this condition, it follows immediately that rank$(W) = m_1$ *necessarily* has to hold. In addition, choosing

$$\widehat{\boldsymbol{z}} = -\sum_{i=1}^{m_1} W_{:,i} \in \mathbb{R}^{m_1},$$

the second-stage constraints

$$W\boldsymbol{y} = \widehat{\boldsymbol{z}},\ \boldsymbol{y} \geq \mathbf{0},$$

by the property of the complete recourse matrix W, are guaranteed to have at

least one feasible solution, say, $\widehat{\boldsymbol{y}}$, such that

$$\begin{aligned}\sum_{i=1}^{m_1} W_{\cdot,i}\widehat{y}_i + \sum_{i=m_1+1}^{\overline{n}} W_{\cdot,i}\widehat{y}_i &= \widehat{\boldsymbol{z}} \\ &= -\sum_{i=1}^{m_1} W_{\cdot,i}, \\ \widehat{y}_i \geq 0,\ i &= 1,\ldots,\overline{n}.\end{aligned}$$

Thus, with

$$\widetilde{y}_i = \begin{cases} \widehat{y}_i + 1 & i = 1,\ldots,m_1, \\ \widehat{y}_i & i = m_1+1,\ldots,\overline{n}, \end{cases}$$

the conditions (2.24) are *necessarily feasible*.

To show *sufficiency*, let us choose an arbitrary $\overline{\boldsymbol{z}} \in \mathbb{R}^{m_1}$. Since the columns $W_{\cdot,1},\ldots,W_{\cdot,m_1}$ are linearly independent, the system of linear equations

$$\sum_{i=1}^{m_1} W_{\cdot,i} y_i = \overline{\boldsymbol{z}},$$

has a unique solution $\overline{y}_1,\ldots,\overline{y}_{m_1}$. If $\overline{y}_i \geq 0$, $i = 1,\ldots,m_1$, we are done. Otherwise, define

$$\gamma := \min\{\overline{y}_1,\ldots,\overline{y}_{m_1}\}.$$

Then $\boldsymbol{y}^*$ defined by

$$y_i^* = \begin{cases} \overline{y}_i - \gamma\widetilde{y}_i & i = 1,\ldots,m_1, \\ -\gamma\widetilde{y}_i & i = m_1+1,\ldots,\overline{n}, \end{cases}$$

solves $W\boldsymbol{y} = \overline{\boldsymbol{z}}$, $\boldsymbol{y} \geq \boldsymbol{0}$. □

Defn 2.8. *If the program* (2.20) *is feasible for all* $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$ *and at least all* $\boldsymbol{x} \in \mathscr{X} = \{\boldsymbol{x} | A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$, *then* (2.17) *is said to be of* ***relatively complete recourse***.

§2.5 CHANCE CONSTRAINED DEP

We will now consider another class of DEPs for (2.5), called **chance constrained stochastic programs**.

Consider a DEP with a collection of constraints

$$g_i(\boldsymbol{x},\boldsymbol{\zeta}) \leq 0,\ i = 1,\ldots,m, \text{ for } \boldsymbol{x} \in \mathscr{X} \subset \mathbb{R}^n.$$

For each realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$ and for some $\alpha \in [0,1]$, we define a *payoff* function for all constraints as

$$\varphi(\boldsymbol{x},\boldsymbol{\varepsilon}) := \begin{cases} 1-\alpha & \text{if } g_i(\boldsymbol{x},\boldsymbol{\varepsilon}) \leq 0,\ \forall i = 1,\ldots,m, \\ -\alpha & \text{otherwise.} \end{cases} \tag{2.25}$$

Consequently, with each realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$, for $\boldsymbol{x}$ infeasible we have an *absolute loss* of α, whereas for $\boldsymbol{x}$ feasible we have *reward* of $1-\alpha$. With this payoff function, it is natural to aim for decisions on $\boldsymbol{x}$ that, at least on average, avoid an absolute loss. That is, we could specify the requirement

$$\mathbf{E}_{\boldsymbol{\zeta}}[\varphi(\boldsymbol{x},\boldsymbol{\zeta})] \geq 0.$$

Define

$$\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon}) := (g_1(\boldsymbol{x},\boldsymbol{\varepsilon}),\ldots,g_m(\boldsymbol{x},\boldsymbol{\varepsilon}))^\top.$$

Note that we write '$\boldsymbol{m} \leq \mathbf{0}$' to indicate that every component of a vector $\boldsymbol{m}$ is less than or equal to 0, and '$\boldsymbol{m} \not\leq \mathbf{0}$' to indicate that *at least one* component of vector $\boldsymbol{m}$ is *not* less than or equal to 0.

If $\boldsymbol{\zeta}$ is a **discrete** random vector, we can write

$$\begin{aligned}
\mathbf{E}_{\boldsymbol{\zeta}}[\varphi(\boldsymbol{x},\boldsymbol{\zeta})] &= \sum_{\boldsymbol{\varepsilon}\in\mathscr{S}_{\boldsymbol{\zeta}}} \varphi(\boldsymbol{x},\boldsymbol{\varepsilon})P(\boldsymbol{\varepsilon}) \\
&= \left(\sum_{\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}} (1-\alpha)P(\boldsymbol{\varepsilon})\right) - \left(\sum_{\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\not\leq\mathbf{0}\}} \alpha P(\boldsymbol{\varepsilon})\right) \\
&= (1-\alpha)\left(\sum_{\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}} P(\boldsymbol{\varepsilon})\right) - \alpha\left(\sum_{\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\not\leq\mathbf{0}\}} P(\boldsymbol{\varepsilon})\right) \\
&= (1-\alpha)P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) - \alpha P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\not\leq\mathbf{0}\}) \\
&= P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) - \alpha\left\{P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq 0\}) + P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\not\leq\mathbf{0}\})\right\} \\
&= P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) - \alpha.
\end{aligned}$$

Therefore, the condition $\mathbf{E}_{\boldsymbol{\zeta}}[\varphi(\boldsymbol{x},\boldsymbol{\zeta})] \geq 0$ is equivalent to requiring

$$P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) \geq \alpha.$$

Note that the same applies when $\boldsymbol{\zeta}$ is a **continuous** random vector. In particular, we can write

$$\begin{aligned}
\mathbf{E}_{\boldsymbol{\zeta}}[\varphi(\boldsymbol{x},\boldsymbol{\zeta})] &= \int_{\mathscr{S}_{\boldsymbol{\zeta}}} \varphi(\boldsymbol{x},\boldsymbol{\varepsilon})dP(\boldsymbol{\varepsilon}) \\
&= \int_{\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}} (1-\alpha)dP(\boldsymbol{\varepsilon}) - \int_{\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\not\leq\mathbf{0}\}} \alpha dP(\boldsymbol{\varepsilon}) \\
&= (1-\alpha)P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) - \alpha P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\not\leq\mathbf{0}\}) \\
&= P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) - \alpha,
\end{aligned}$$

which would lead us to the same conclusion, that $\mathbf{E}_{\boldsymbol{\zeta}}[\varphi(\boldsymbol{x},\boldsymbol{\zeta})] \geq 0$ is equivalent to requiring

$$P(\{\boldsymbol{\varepsilon}:\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})\leq\mathbf{0}\}) \geq \alpha.$$

Defn 2.9. *A **probabilistically constrained program**, also called a **chance constrained program** or **a problem with joint probabilistic constraints**, is defined as*

$$\begin{aligned} \min \quad & w = \mathbf{E}_{\boldsymbol{\zeta}}[f_0(x,\boldsymbol{\zeta})] \\ \textit{such that} \quad & P\left(\{\boldsymbol{\varepsilon} : g_i(\boldsymbol{x},\boldsymbol{\varepsilon}) \le 0,\ i = 1,\dots,m\}\right) \ge \alpha. \end{aligned} \tag{2.26}$$

Instead of the payoff function (2.25), we might choose to define $\alpha_i \in [0,1]$, $i = 1,\dots,m$, and analogous payoffs for every single constraint, thus allowing us to weigh certain constraints as more important than others. This results in

$$\varphi_i(\boldsymbol{x},\boldsymbol{\varepsilon}) := \begin{cases} 1-\alpha_i & \text{if } g_i(\boldsymbol{x},\boldsymbol{\varepsilon}) \le 0, \\ \alpha_i & \text{otherwise,} \end{cases} \tag{2.27}$$

for $i = 1,\dots,m$. Then we get the following DEP.

Defn 2.10. *A **single probabilistic/chance constrained problem**, or **separate probabilistic/chance constrained problem**, is defined as follows:*

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathscr{X}} \quad & w = \mathbf{E}_{\boldsymbol{\zeta}}[f_0(\boldsymbol{x},\boldsymbol{\zeta})] \\ \textit{such that} \quad & P\left(\{\boldsymbol{\varepsilon} : g_i(\boldsymbol{x},\boldsymbol{\zeta}) \le 0\}\right) \ge \alpha_i,\ i = 1,\dots,m. \end{aligned} \tag{2.28}$$

There are *many* other possibilities to generate types of DEPs for (2.5) by constructing the payoff functions φ_i in different ways, out of the objective function and the constraints. All the problems derived here are *mathematical programs*.

CHANCE CONSTRAINED DEP: PROPERTIES

For chance constrained programs, things are generally more difficult than for recourse problems. Consider the constraint

$$P\left(\{\boldsymbol{\varepsilon} : \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\}\right) \geq \alpha$$

of (2.26) with the functions g_i replaced by the vector-valued function

$$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\varepsilon}) := (g_1(\boldsymbol{x}, \boldsymbol{\varepsilon}), \ldots, g_m(x, \boldsymbol{\varepsilon}))^\top.$$

A point $\widehat{\boldsymbol{x}}$ is feasible iff

$$P(\{\boldsymbol{\varepsilon} : \boldsymbol{g}(\widehat{\boldsymbol{x}}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\}) \geq \alpha;$$

that is, $\widehat{\boldsymbol{x}}$ is feasible iff the set (event)

$$\mathscr{S}(\widehat{\boldsymbol{x}}) = \{\boldsymbol{\varepsilon} : \boldsymbol{g}(\widehat{\boldsymbol{x}}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\}$$

has a probability measure of at least α.

Another way to define a feasible point $\widehat{\boldsymbol{x}}$ is as follows. Let $\mathbb{G}$ be a collection of all events such that

$$P(G) \geq \alpha \text{ for all } G \in \mathbb{G}.$$

Then, $\widehat{\boldsymbol{x}}$ is feasible iff we have at least one event $\widetilde{G} \in \mathbb{G}$ such that for all $\boldsymbol{\varepsilon} \in \widetilde{G}$, $\boldsymbol{g}(\widehat{\boldsymbol{x}}, \boldsymbol{\varepsilon}) \leq \mathbf{0}$. Formally, $\widehat{\boldsymbol{x}}$ is feasible iff there exists $\widetilde{G} \in \mathbb{G}$ such that

$$\widehat{\boldsymbol{x}} \in \bigcap_{\boldsymbol{\varepsilon} \in \widetilde{G}} \{\boldsymbol{x} : \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\}. \tag{2.29}$$

Hence, the feasible set

$$B(\alpha) = \left\{\boldsymbol{x} : P\left(\{\boldsymbol{\varepsilon} : \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\}\right) \geq \alpha\right\}$$

is the union of all those vectors $\boldsymbol{x}$ feasible according to (2.29). Consequently, $B(\alpha)$ may be written as

$$B(\alpha) = \bigcup_{G \in \mathbb{G}} \bigcap_{\boldsymbol{\varepsilon} \in G} \left\{\boldsymbol{x} : \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\right\}.$$

Since a union of convex sets is not necessarily convex, this demonstrates that in general we may not expect $B(\alpha)$ to be convex, even if $\{\boldsymbol{x} : \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\varepsilon}) \leq \mathbf{0}\}$ is convex for all $\boldsymbol{\varepsilon} \in \mathscr{S}_{\boldsymbol{\zeta}}$. Indeed, there are simple examples of nonconvex feasible sets.

***Example* 28.** Assume that in our refinery example demands are random with the following joint discrete distribution:

$$P(h_s(\boldsymbol{\varepsilon}_1) = 1600,\, h_p(\boldsymbol{\varepsilon}_1) = 1350) = 0.85,$$
$$P(h_s(\boldsymbol{\varepsilon}_2) = 1500,\, h_p(\boldsymbol{\varepsilon}_2) = 1950) = 0.08,$$
$$P(h_s(\boldsymbol{\varepsilon}_3) = 2000,\, h_p(\boldsymbol{\varepsilon}_3) = 1200) = 0.07.$$

Recall the constraints $x_1 + x_2 \leq 100, x_1, x_2 \geq 0$, but this time with the probabilistic constraints

$$P(20x_1 + 60x_2 \geq h_s(\boldsymbol{\varepsilon}),\, 30x_1 + 30x_2 \geq h_p(\boldsymbol{\varepsilon})) \geq \alpha,$$

for any $\alpha \in (0.85, 0.92]$. These constraints, combined together, require that we either

1. either satisfy the demands $h_s(\varepsilon_1)$ and $h_p(\varepsilon_2)$, thus enforcing a reliability of 93%, and hence choose a production program to cover a demand $h_a = (1600; 1950)$;

2. or satisfy the demands $h_s(\varepsilon_3)$ and $h_p(\varepsilon_1)$, thus enforcing a reliability of 92%, and hence choose a production program to cover a demand $h_b = (2000; 1350)$.

The above scenarios each has a convex solution space, but the feasible set for the union of the above constraints can clearly be seen to be nonconvex in Figure 2.8.

A natural question arises: when is $B(\alpha)$ convex? The following proposition addresses this question.

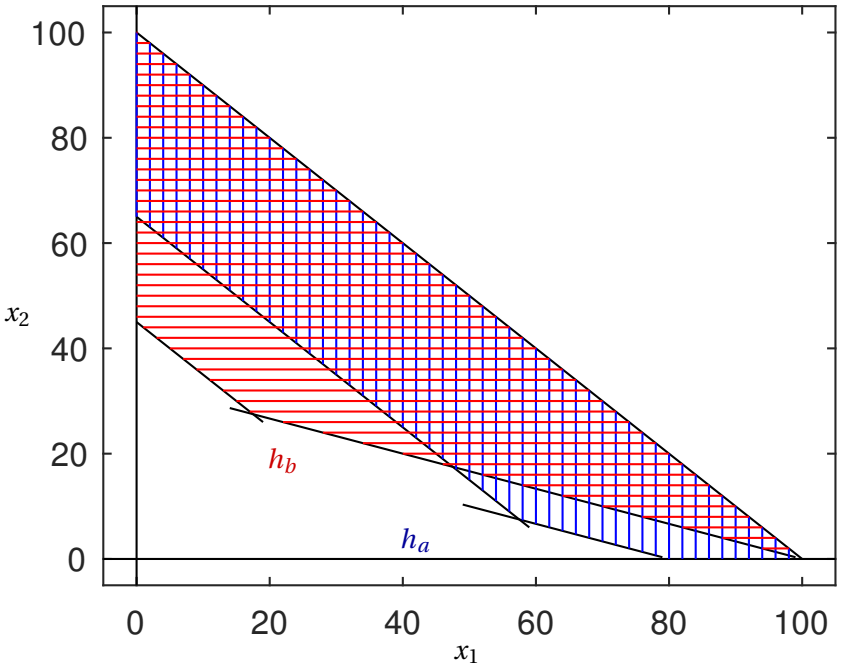


Figure 2.8: *Chance constraints can lead to a non-convex feasible set*

Proposition 2.3. *If $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})$ is jointly convex and the probability measure P is quasi-concave, then for all $\alpha \in [0,1]$, the feasible set*

$$B(\alpha) = \left\{\boldsymbol{x} : P(\{\boldsymbol{\varepsilon} : \boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon}) \leq \boldsymbol{0}\}) \geq \alpha\right\}$$

is convex.

Before we go into the proof, we need to define the concept of a probability measure P being quasi-concave. Suppose that the random vector $\boldsymbol{\zeta}$ lives on the probability space $(\Omega, \mathbb{F}, P)$, so $\mathbb{F}$ is the σ-field on which P is defined. Then,

P is said to be quasi-concave if

$$P(\lambda\mathscr{S}_1 + (1-\lambda)\mathscr{S}_2) \geq \min\{P(\mathscr{S}_1),\, P(\mathscr{S}_2)\} \quad \text{for all } \lambda \in (0,1)$$

for all convex sets $\mathscr{S}_i \in \mathbb{F}$, $i = 1,2$.

Proof. Define

$$\mathscr{S}(\boldsymbol{x}) := \left\{\boldsymbol{\varepsilon} : \boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon}) \leq \mathbf{0}\right\}.$$

For $\boldsymbol{x}_i, \boldsymbol{x}_2 \in B(\alpha)$, $\boldsymbol{\varepsilon}_1 \in \mathscr{S}(\boldsymbol{x}_1)$, $\boldsymbol{\varepsilon}_2 \in \mathscr{S}(\boldsymbol{x}_2)$, and $\lambda \in (0,1)$, define

$$(\overline{\boldsymbol{x}}, \overline{\boldsymbol{\varepsilon}}) := \lambda(\boldsymbol{x}_1, \boldsymbol{\varepsilon}_1) + (1-\lambda)(\boldsymbol{x}_2, \boldsymbol{\varepsilon}_2);$$

In other words, define $\overline{\boldsymbol{x}} := \lambda\boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2$ and $\overline{\boldsymbol{\varepsilon}} := \lambda\boldsymbol{\varepsilon}_1 + (1-\lambda)\boldsymbol{\varepsilon}_2$. If $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})$ is jointly convex, then it follows that

$$\boldsymbol{g}(\overline{\boldsymbol{x}}, \overline{\boldsymbol{\varepsilon}}) \leq \lambda\boldsymbol{g}(\boldsymbol{x}_1, \boldsymbol{\varepsilon}_1) + (1-\lambda)\boldsymbol{g}(\boldsymbol{x}_2, \boldsymbol{\varepsilon}_2) \leq 0,$$

where the second inequality is due to the assumptions that $\boldsymbol{\varepsilon}_1 \in \mathscr{S}(\boldsymbol{x}_1)$ and $\boldsymbol{\varepsilon}_2 \in \mathscr{S}(\boldsymbol{x}_2)$. The inequality $\boldsymbol{g}(\overline{\boldsymbol{x}}, \overline{\boldsymbol{\varepsilon}}) \leq \mathbf{0}$ implies $\overline{\boldsymbol{\varepsilon}} \in \mathscr{S}(\overline{\boldsymbol{x}})$, and hence

$$\lambda\mathscr{S}(\boldsymbol{x}_1) + (1-\lambda)\mathscr{S}(\boldsymbol{x}_2) \subset \mathscr{S}(\overline{\boldsymbol{x}}), \tag{2.30}$$

where the addition of two sets $\mathscr{A}$ and $\mathscr{B}$ is performed as follows:

$$\rho\mathscr{A} + \sigma\mathscr{B} := \{\boldsymbol{\varepsilon} := \rho\boldsymbol{\varepsilon}_1 + \sigma\boldsymbol{\varepsilon}_2 : \boldsymbol{\varepsilon}_1 \in \mathscr{A},\, \boldsymbol{\varepsilon}_2 \in \mathscr{B}\}.$$

Equation (2.30) implies

$$P(\mathscr{S}(\overline{\boldsymbol{x}})) \geq P(\lambda\mathscr{S}(\boldsymbol{x}_1) + (1-\lambda)\mathscr{S}(\boldsymbol{x}_2)).$$

By the joint convexity on $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\varepsilon})$, any set $\mathscr{S}(\boldsymbol{x})$ is convex. (For any $\boldsymbol{x}$, take $\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2 \in \mathscr{S}(\boldsymbol{x})$ and show that $\lambda\boldsymbol{\varepsilon}_1 + (1-\lambda)\boldsymbol{\varepsilon}_2 \in \mathscr{S}(\boldsymbol{x})$.) As by assumption P is quasi-concave, we have

$$P(\lambda\mathscr{S}(\boldsymbol{x}_1) + (1-\lambda)\mathscr{S}(\boldsymbol{x}_2)) \geq \min\{P(\mathscr{S}(\boldsymbol{x}_1)),\, P(\mathscr{S}(\boldsymbol{x}_2))\} \geq \alpha \quad \text{for all } \lambda \in (0,1)$$

for all convex sets $\mathscr{S}_i \in \mathbb{F}$, $i = 1,2$.

Thus, $P(\mathscr{S}(\overline{\boldsymbol{x}})) \geq \alpha$, which implies $\overline{\boldsymbol{x}} \in B(\alpha)$ and therefore $B(\alpha)$ is convex for all $\alpha \in (0,1)$ □

Therefore, we see that P being quasi-concave assists us in establishing a nice property of chance constrained DEPs. Hence, we would like to be able to classify when a probability measure P is quasi-concave.

Theorem 2.2. *If the probability measure P is quasi-concave, then the corresponding distribution function $F_{\boldsymbol{\zeta}}$ is quasi-concave.*

This follows from the definition of distribution functions:

$$F_{\boldsymbol{\zeta}}(\boldsymbol{\varepsilon}_i) = P(\mathscr{S}_i)$$

with $\mathscr{S}_i = \{\boldsymbol{\varepsilon} : \boldsymbol{\varepsilon} \leq \boldsymbol{\varepsilon}_i\}$, $i = 1,2$, and from the fact that for $\widehat{\boldsymbol{\varepsilon}} = \lambda\boldsymbol{\varepsilon}_1 + (1-\lambda)\boldsymbol{\varepsilon}_2$, $\lambda \in (0,1)$ we have $\widehat{\mathscr{S}} = \{\boldsymbol{\varepsilon} : \boldsymbol{\varepsilon} \leq \widehat{\boldsymbol{\varepsilon}}\} = \lambda\mathscr{S}_1 + (1-\lambda)\mathscr{S}_2$.

When P is quasi-concave, this gives

$$F_{\boldsymbol{\zeta}}(\widehat{\boldsymbol{\varepsilon}}) = P(\widehat{\mathscr{S}}) \geq \min\{P(\mathscr{S}_1), P(\mathscr{S}_2)\} = \min\left\{F_{\boldsymbol{\zeta}}(\boldsymbol{\varepsilon}_1), F_{\boldsymbol{\zeta}}(\boldsymbol{\varepsilon}_2)\right\}.$$

Unfortunately, the converse is not true: $F_{\boldsymbol{\zeta}}$ being quasi-concave does not imply, in general, that the corresponding probability measure P is quasi-concave. **[Example in class]**

§2.6 Lagrange Multipliers, LPs, and Duality

Before looking at the Dual Decomposition Method, We will revisit the relationship between Linear Programs, Duality and Lagrange multipliers. **Lagrange multipliers** provide another method for solving constrained optimisation problems and because of this, We will discover a direct relationship between them and the Dual of an LP.

***Example* 29.** Consider the following non-linear optimisation problem:

$$\begin{aligned} \min \quad & f(x,y) = x^2 + y^2 \\ \text{such that} \quad & g(x,y) = x + y - 2 = 0. \end{aligned} \tag{2.31}$$

The solution is clearly $x = y = 1$ with $f(x, y) = 2$.

The contour of the objective function $f(x, y)$ is tangent to the constraint surface at the solution, as can be seen in Figure 2.9.

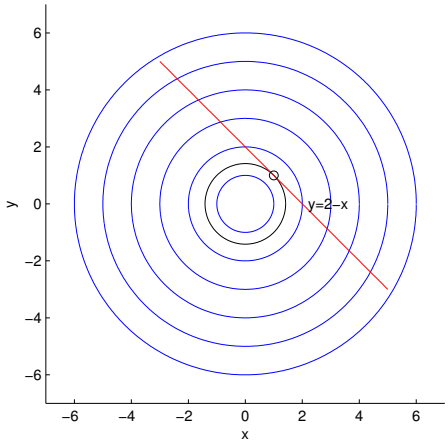


Figure 2.9: *Contours of the objective function and constraint*

The simplest version of the **Lagrange Multiplier Theorem** says that this is always true with equality constraints: *at the constraint optimum, if it exists,* ∇f *will be a multiple of* ∇g.

The Lagrange Multiplier Theorem enables us translate the original constrained optimisation problem into an ordinary system of simultaneous equations at the cost of introducing a new variable; that is,

$$g(x, y) = 0, \tag{2.32}$$

$$\nabla f(x, y) = -p\nabla g(x, y), \tag{2.33}$$

where Equation (2.32) says we must satisfy the original constraint (2.31), and Equation (2.33) adds that ∇f must be parallel to ∇g. The new variable p is called the **Lagrange multiplier**.

This system can be written more compactly by writing it in terms of the Lagrangian L, where

$$\begin{aligned} L(x,y,p) &= f(x,y) + pg(x,y) \\ &= x^2 + y^2 + p(x+y-2). \end{aligned}$$

We then set $\nabla L = \mathbf{0}$, that is,

$$\begin{aligned} \nabla_p L(x,y,p) &= x+y-2 = 0, \\ \nabla_x L(x,y,p) &= 2x+p = 0, \\ \nabla_y L(x,y,p) &= 2y+p = 0. \end{aligned}$$

The unique solution of the above system of equations is $x = y = 1$ as before, with $p = -2$.

This technique applies to more than problems with single constraints, by introducing more than one Lagrange multiplier.

***Example* 30.** Consider the following optimisation.

$$\begin{aligned} \min \quad & f(x,y,z) = x^2 + y^2 + z^2, \\ \text{such that} \quad & x+y-2 = 0, \\ & x+z-2 = 0. \end{aligned}$$

Then, $L(x,y,z,p,q) = x^2 + y^2 + z^2 + p(x+y-2) + q(x+z-2)$, and setting $\nabla L = \mathbf{0}$ gives

$$\begin{aligned} \nabla_p L(x,y,z,p,q) &= x+y-2 = 0, \\ \nabla_q L(x,y,z,p,q) &= x+z-2 = 0, \\ \nabla_x L(x,y,z,p,q) &= 2x+p+q = 0, \\ \nabla_y L(x,y,z,p,q) &= 2y+p = 0, \\ \nabla_z L(x,y,z,p,q) &= 2z+q = 0. \end{aligned}$$

The unique solution, which is a little more difficult to find here, is

$$x = \frac{4}{3}, y = z = \frac{2}{3}, p = q = -\frac{4}{3}.$$

The level surface of the objective function $f(x, y, z)$ is a sphere, and the constraints are no longer tangent to the level surface at the solution (see Figure 2.10).

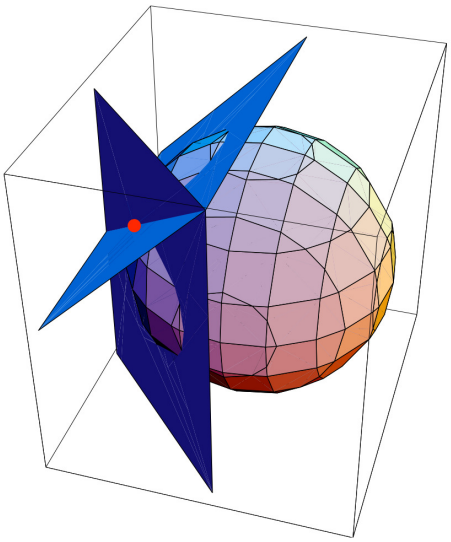


Figure 2.10: *Objective function (sphere) and two constraints (planes)*

The normal to the surface is however a linear combination of the normals to the two constraint surfaces, with coefficients p and q.

This is an illustration of another version of the **Lagrange Multiplier Theorem**, which says that *if a solution exists, it is a saddle point of the Lagrangian such that no change in the original variables can decrease the Lagrangian, while no change in the Lagrange multipliers can increase it.*

What about inequality constraints?

***Example* 31.**

$$\begin{aligned} \min \;\; & f(x,y) = x^2 + y^2 \\ \text{such that} \;\; & x + y - 2 \geq 0. \end{aligned}$$

Here, $L(x,y,p) = x^2 + y^2 + p(x+y-2)$, but now $p \leq 0$. Unfortunately when there are inequality constraints, $\nabla L = \mathbf{0}$ is neither necessary or sufficient to guarantee a solution to a constrained problem. We need the following theorem.

Theorem 2.3 (Lagrange multipliers)**.** *Solving*

$$\begin{aligned} \min \;\; & f(\boldsymbol{x}) \\ \textit{such that} \;\; & A\boldsymbol{x} + \boldsymbol{b} \leq \mathbf{0} \end{aligned}$$

is equivalent to solving

$$\min_{\boldsymbol{x}} \left(\max_{\boldsymbol{p} \in \mathbb{R}^n_+} L(\boldsymbol{x}, \boldsymbol{p}) \right),$$

where ***the Lagrangian*** *is defined as*

$$L(\boldsymbol{x}, \boldsymbol{p}) = f(\boldsymbol{x}) + \boldsymbol{p}^\top (A\boldsymbol{x} + \boldsymbol{b}),$$

for a vector of ***Lagrange multipliers*** $\boldsymbol{p}$*.*

This theorem uses properties of convex cones and duality to transform the original problem to a problem that uses only the very simple cone $\mathbb{R}^n_+$.

***Example* 32.** Consider the following linear program.

$$\left.\begin{aligned} \min \;\; & f(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{x} \\ \text{such that} \;\; & A\boldsymbol{x} + \boldsymbol{b} \leq \mathbf{0} \\ & \boldsymbol{x} \geq \mathbf{0} \end{aligned}\right\} \tag{2.34}$$

with

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{p}) &= \boldsymbol{c}^\top \boldsymbol{x} + \boldsymbol{p}^\top \left(A\boldsymbol{x} + \boldsymbol{b}\right) \\ &= \left(\boldsymbol{c}^\top + \boldsymbol{p}^\top A\right)\boldsymbol{x} + \boldsymbol{p}^\top \boldsymbol{b}. \end{aligned} \tag{2.35}$$

Note that (2.35) is the Lagrangian for the dual LP of (2.34). and the Lagrange multipliers are in fact dual variables.

That this is true becomes useful as many of the algorithms for solution of Dual Decomposition problems are written in terms of Lagrange multipliers. These may be simply found in MATLAB as the solution of the Dual of the associated LPs.

§2.7 Dual Decomposition Method

We now return to the DEP given by (2.17):

$$\begin{aligned} \min \quad & w = \boldsymbol{c}^\top \boldsymbol{x} + \mathbf{E}_{\boldsymbol{\zeta}}[Q(\boldsymbol{x},\boldsymbol{\zeta})] \\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \mathbf{0}, \end{aligned}$$

where for each realisation $\boldsymbol{\varepsilon}$ of $\boldsymbol{\zeta}$,

$$Q(\boldsymbol{x},\boldsymbol{\varepsilon}) = \min_{\boldsymbol{y}} \left\{ \boldsymbol{q}^\top \boldsymbol{y} \,\middle|\, W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x},\ \boldsymbol{y} \geq \mathbf{0} \right\}$$

or, equivalently,

$$\begin{aligned} \min \quad & \boldsymbol{q}^\top \boldsymbol{y} \\ \text{such that} \quad & W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}) - T(\boldsymbol{\varepsilon})\boldsymbol{x} \\ & \boldsymbol{y} \geq \mathbf{0}. \end{aligned}$$

This DEP has the so-called **Dual Decomposition Structure**:

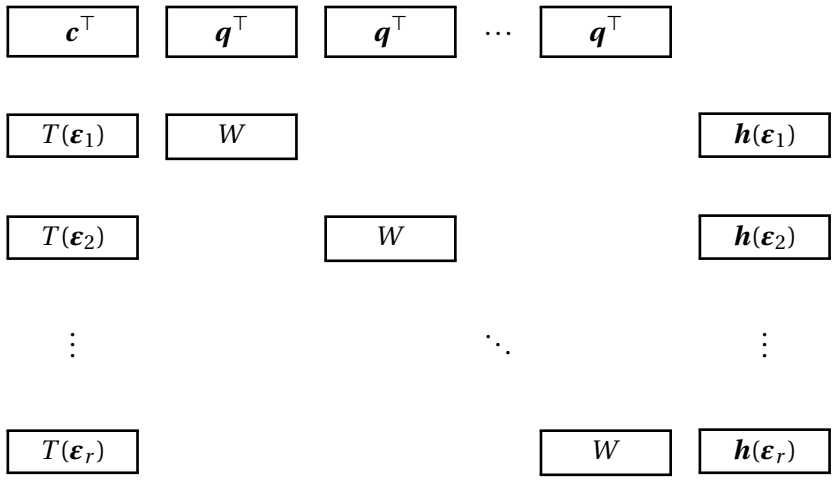


Figure 2.11: *Dual Decomposition Structure*

We will now consider a solution method that exploits this structure. For simplicity and clarity of exposition, we restrict our attention to the case where S_ζ contains just one realisation (the random vector $\boldsymbol{\zeta}$ has just one realisation); hence, the problem reduces to the LP:

$$\left.\begin{array}{rl} \min\limits_{\boldsymbol{x}\in\mathscr{X}} & z = \boldsymbol{c}^\top\boldsymbol{x} + \boldsymbol{q}^\top\boldsymbol{y} \\ \text{such that} & A\boldsymbol{x} = \boldsymbol{b}, \\ & T\boldsymbol{x} + W\boldsymbol{y} = \boldsymbol{h}, \\ & \boldsymbol{x}, \boldsymbol{y} \geq \boldsymbol{0}. \end{array}\right\} \tag{2.36}$$

Furthermore, we will assume that the problem is *solvable* and that the first-stage feasible set $\{\boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}\}$ is bounded. Then, the LP (2.36) may

be restated as a non-linear program:

$$\left.\begin{array}{rl} \min & z = \boldsymbol{c}^\top \boldsymbol{x} + Q(\boldsymbol{x}) \\ \text{such that} & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0}, \end{array}\right\} \tag{2.37}$$

with

$$Q(\boldsymbol{x}) := \min_{\boldsymbol{y}} \{\boldsymbol{q}^\top \boldsymbol{y} : W\boldsymbol{y} = \boldsymbol{h} - T\boldsymbol{x}, \boldsymbol{y} \geq \boldsymbol{0}\}.$$

The recourse function $Q(\boldsymbol{x})$ is piecewise linear and convex. The problem (2.37) is also equivalent to the problem

$$\left.\begin{array}{rl} \min & z = \boldsymbol{c}^\top \boldsymbol{x} + \theta \\ \text{such that} & A\boldsymbol{x} = \boldsymbol{b} \\ & \theta - Q(\boldsymbol{x}) \geq 0 \\ & \boldsymbol{x} \geq \boldsymbol{0}, \end{array}\right\} \tag{2.38}$$

where the minimisation is over $\boldsymbol{x}$ and $\theta \in \mathbb{R}$. However, (2.38) requires knowledge of the recourse function $Q(\boldsymbol{x})$ in advance. Such knowledge will not be common in practice; that is, an explicit form will typically not be available.

Therefore, we might try to construct a sequence of additional linear constraints that can be used to define a monotonically decreasing feasible set $\mathscr{B}_1$ of $(n+1)$-vectors, $(x_1, x_2, \ldots, x_n, \theta)^\top$, such that eventually, with

$$\mathscr{B}_0 := \{(\boldsymbol{x}^\top, \theta)^\top : A\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0},\ \theta \in \mathbb{R}\}$$

the problem

$$\min_{(\boldsymbol{x},\boldsymbol{\theta}) \in \mathscr{B}_0 \cap \mathscr{B}_1} \boldsymbol{c}^\top \boldsymbol{x} + \theta$$

yields a first-stage solution $\boldsymbol{x}$ of (2.36).

Dual Decomposition Method

Step 1. Let θ_0 be a lower bound[1] for

$$\min_{\boldsymbol{y}} \left\{ \boldsymbol{q}^\top \boldsymbol{y} : A\boldsymbol{x} = \boldsymbol{b},\ T\boldsymbol{x} + W\boldsymbol{y} = \boldsymbol{h},\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{y} \geq \boldsymbol{0} \right\}.$$

Solve the linear program

$$\min_{\boldsymbol{x},\theta} \left\{ \boldsymbol{c}^\top \boldsymbol{x} + \theta : A\boldsymbol{x} = \boldsymbol{b},\ \theta \geq \theta_0,\ \boldsymbol{x} \geq \boldsymbol{0} \right\},$$

and denote its solution[2] $(\widehat{\boldsymbol{x}}, \widehat{\theta})$. Let $\mathscr{B}_1 := \left\{ \mathbb{R}^n \times \{\theta\} : \theta \geq \theta_0 \right\}$.

Step 2. Using the last first-stage solution $\widehat{\boldsymbol{x}}$, evaluate the recourse function (i.e. solve the following LP):

$$\begin{aligned} Q(\widehat{\boldsymbol{x}}) &= \min \left\{ \boldsymbol{q}^\top \boldsymbol{y} : W\boldsymbol{y} = \boldsymbol{h} - T\widehat{\boldsymbol{x}},\ \boldsymbol{y} \geq \boldsymbol{0} \right\} \\ &= \max \left\{ (\boldsymbol{h} - T\widehat{\boldsymbol{x}})^\top \boldsymbol{u} : W^\top \boldsymbol{u} \leq \boldsymbol{q},\ \boldsymbol{u} \text{ free} \right\}. \end{aligned}$$

where the second equality follows from *duality*.

Two cases may arise: (a) $Q(\widehat{\boldsymbol{x}})$ is unbounded, or (b) $Q(\widehat{\boldsymbol{x}})$ is bounded.

a) If $Q(\widehat{\boldsymbol{x}})$ is unbounded, then $\widehat{\boldsymbol{x}}$ is not feasible with respect to all the constraints of (2.36) (i.e. $\widehat{\boldsymbol{x}}$ does not satisfy the induced constraints, as discussed in Theorem 2.1), for any values of $\boldsymbol{q} \geq \boldsymbol{0}$.

Then, we have a $\widetilde{\boldsymbol{u}}$ such that

$$\begin{aligned} &W^\top \widetilde{\boldsymbol{u}} \leq \boldsymbol{0} \quad \text{(suppose we choose } \boldsymbol{q} = \boldsymbol{0}\text{)}, \\ &(\boldsymbol{h} - T\widehat{\boldsymbol{x}})^\top \widetilde{\boldsymbol{u}} > 0. \end{aligned}$$

[1] How do we know that θ_0 exists? Because of the assumption that the original problem (2.36) is solvable; that is, there exists at least one solution to the first-stage problem, $\boldsymbol{x}$, such that $Q(\boldsymbol{x})$ has a solution.

[2] The existence of this solution is due to the second assumption: that the feasible region of the first-stage problem, $\{\boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$, is bounded.

However, for any $\boldsymbol{x}$ that is feasible to the original SLP (2.36), there must exist a $\boldsymbol{y} \geq \boldsymbol{0}$ such that $W\boldsymbol{y} = \boldsymbol{h} - T\boldsymbol{x}$, and therefore scalar multiplication of this equation by $\tilde{\boldsymbol{u}}$ yields

$$\tilde{\boldsymbol{u}}^\top(\boldsymbol{h} - T\boldsymbol{x}) = \underbrace{\tilde{\boldsymbol{u}}^\top W}_{\leq 0}\underbrace{\boldsymbol{y}}_{\geq 0} \leq 0, \tag{2.39}$$

which in turn implies that $\tilde{\boldsymbol{u}}^\top \boldsymbol{h} \leq \tilde{\boldsymbol{u}}^\top T\boldsymbol{x}$ must also hold for any feasible $\boldsymbol{x}$. (Inequality (2.39) obviously does not hold for $\hat{\boldsymbol{x}}$, since, as noted above, $\tilde{\boldsymbol{u}}^\top(\boldsymbol{h} - T\hat{\boldsymbol{x}}) > 0$.) Therefore, we introduce the **feasibility cut**, cutting off the infeasible solution $\hat{\boldsymbol{x}}$:

$$\tilde{\boldsymbol{u}}^\top(\boldsymbol{h} - T\boldsymbol{x}) \leq 0.$$

Then, redefine

$$\mathscr{B}_1 := \mathscr{B}_1 \cap \left\{(\boldsymbol{x}, \theta) : \tilde{\boldsymbol{u}}^\top(\boldsymbol{h} - T\boldsymbol{x}) \leq 0\right\}$$

and proceed to **Step 3**.

b) Otherwise, if $Q(\hat{\boldsymbol{x}})$ is bounded, we have for the recourse problem, simultaneously, a primal optimal solution $\hat{\boldsymbol{y}}$ and a dual optimal solution $\hat{\boldsymbol{u}}$ (with $W^T\hat{u} \leq q$). From the dual formulation of the recourse problem, we have

$$Q(\hat{\boldsymbol{x}}) = (\boldsymbol{h} - T\hat{\boldsymbol{x}})^\top \hat{\boldsymbol{u}},$$

whereas for any $\boldsymbol{x}$ we have

$$\begin{aligned} Q(\boldsymbol{x}) &= \sup_{\boldsymbol{u}} \left\{(\boldsymbol{h} - T\boldsymbol{x})^\top \boldsymbol{u} : W^\top \boldsymbol{u} \leq \boldsymbol{q}\right\} \\ &\geq (\boldsymbol{h} - T\boldsymbol{x})^\top \hat{\boldsymbol{u}} \\ &= \hat{\boldsymbol{u}}^\top(\boldsymbol{h} - T\boldsymbol{x}). \end{aligned}$$

The intended constraint $\theta \geq Q(\boldsymbol{x})$ in (2.38) implies the linear constraint

$$\theta \geq \hat{\boldsymbol{u}}^\top(\boldsymbol{h} - T\boldsymbol{x}),$$

which is violated by $(\widehat{\boldsymbol{x}},\widehat{\theta})$ if and only if $(\boldsymbol{h}-T\widehat{\boldsymbol{x}})^{\top}\widehat{\boldsymbol{u}} > \widehat{\theta}$; in this case we introduce the **optimality cut**, cutting off the non-optimal solution $(\widehat{\boldsymbol{x}},\widehat{\theta})$:

$$\theta \geq \widehat{\boldsymbol{u}}^{\top}(\boldsymbol{h}-T\boldsymbol{x}).$$

Correspondingly, we redefine

$$\mathscr{B}_1 := \mathscr{B}_1 \cap \left\{(\boldsymbol{x},\theta) : \theta \geq \widehat{\boldsymbol{u}}^{\top}(\boldsymbol{h}-T\boldsymbol{x})\right\},$$

and continue to **Step 3**. Otherwise, if $Q(\widehat{\boldsymbol{x}}) \leq \widehat{\theta}$, we stop.

Step 3. Solve the updated (linear programming) problem

$$\min\left\{\boldsymbol{c}^{\top}\boldsymbol{x}+\theta : (\boldsymbol{x}^{\top},\theta) \in \mathscr{B}_0 \cap \mathscr{B}_1\right\},$$

yielding the optimal solution $(\widetilde{\boldsymbol{x}},\widetilde{\theta})$.

With $(\widehat{\boldsymbol{x}},\widehat{\theta}) := (\widetilde{\boldsymbol{x}},\widetilde{\theta})$, return to **Step 2**.

This has dealt with the scenario where there is only one set of constraints, because of the assumption that the random vector $\boldsymbol{\zeta}$ has only one realisation. In summary, in the Dual Decomposition Method, we first look at the induced constraints problem and make appropriate feasibility cuts to the original first stage LP. Once we have a feasible second stage solution, we optimise that solution, which can also require the inclusion of further constraints in the first stage LP. This process is iterated until (assuming solvability) the optimal solution is determined.

When $\boldsymbol{\zeta}$ has more than one realisation, we develop some pseudo-code, which naturally follows what we have just seen and gives us a handle on these types of problem. The computational burden with respect to the DEP is the solution of all the second-stage recourse linear programs, which essentially depends on the number of realisations of $\boldsymbol{\zeta}$. When this number is a finite value r, with probabilities p_k for $1 \leq k \leq r$, the computational burden can be significantly reduced by associating one set of second-stage decisions $\boldsymbol{y}_k$ to each realisation $\boldsymbol{\varepsilon}_k$.

In other words, consider the expanded form of (2.17) given by

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathscr{X}} \quad & w = \boldsymbol{c}^\top \boldsymbol{x} + \sum_{k=1}^{r} p_k \boldsymbol{q}_k^\top \boldsymbol{y}_k \\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & T_k \boldsymbol{x} + W \boldsymbol{y}_k = \boldsymbol{h}_k \ \text{ for } 1 \le k \le r \\ & \boldsymbol{x}, \boldsymbol{y}_k \ge \boldsymbol{0} \ \text{ for } 1 \le k \le r. \end{aligned}$$

For each realisation $\boldsymbol{\varepsilon}_k$ of $\boldsymbol{\zeta}$, $1 \le k \le r$, we have an extra row in the Decomposition Structure.

Proposition 2.4. *If the program* (2.36) *is solvable and* $\{\boldsymbol{x} : A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \ge \boldsymbol{0}\}$ *is bounded, then the Dual Decomposition Method converges to an optimal solution after finitely many steps.*

We will now introduce the **L-shaped algorithm** for solving stochastic linear programs with the dual decomposition structure. The **L-shaped algorithm** (name derivative of the below structure) takes advantage of this structure, making feasibility and optimality cuts (essentially adding constraints) during an iterative process.

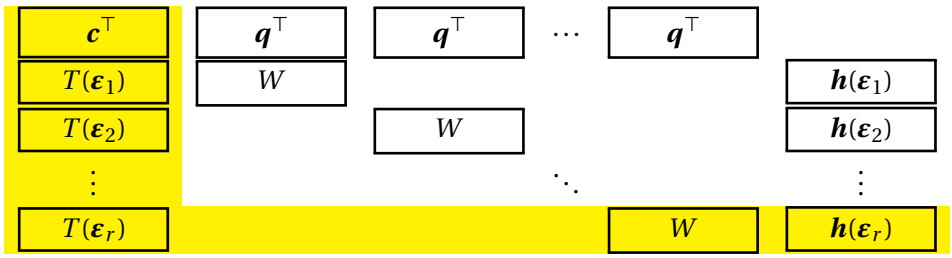


Figure 2.12: *Dual Decomposition Structure*

Initially, we will state the algorithm in pseudo-code and then demonstrate by way of example its processes of creating feasibility and optimality cuts.

The L-shaped Algorithm

Step 0**. Initialisation.** Set

$$t = s = \nu := 0, \quad \boldsymbol{D}_0 := \boldsymbol{0}, \quad d_0 := 0, \quad \boldsymbol{E}_0 := \boldsymbol{0}, \quad e_0 := 0.$$

Here, ν is the counter for iterations, t for the feasibility cuts, and s for optimality cuts.

Step 1**.** Set $\nu = \nu + 1$ and solve the LP

$$\begin{aligned} \min \quad & z = \boldsymbol{c}^\top \boldsymbol{x} + \theta && (2.40)\\ \text{such that} \quad & A\boldsymbol{x} = \boldsymbol{b} && (2.41)\\ & \boldsymbol{D}_\ell \boldsymbol{x} \geq d_\ell \quad \text{for } \ell = 1, \ldots, t && (2.42)\\ & \boldsymbol{E}_\ell \boldsymbol{x} + \theta \geq e_\ell \quad \text{for } \ell = 1, \ldots, s && (2.43)\\ & \boldsymbol{x} \geq \boldsymbol{0}, \ \theta \in \mathbb{R}. && (2.44) \end{aligned}$$

If no constraints (2.43) exist, we set $\theta = 0$, to solve the LP. The optimal solution at iteration ν is denoted $\left(\boldsymbol{x}^{(\nu)}, \theta^{(\nu)}\right)$, where we set $\theta^{(\nu)} = -\infty$ if no constraints (2.43) exist.

Note that the constraints (2.42) correspond to feasibility cuts, and the constraints (2.43) correspond to optimality cuts.

Step 2**. Feasibility cuts.** For $k = 1, \ldots, r$ (consider all realisations of $\boldsymbol{\xi}$) and while $\widetilde{w} > 0$, solve the LP

$$\begin{aligned} \min \quad & \widetilde{w} = \boldsymbol{e}^\top \boldsymbol{v}^+ + \boldsymbol{e}^\top \boldsymbol{v}^- && (2.45)\\ \text{such that} \quad & W\boldsymbol{y}_k^{(\nu)} + \boldsymbol{v}^+ - \boldsymbol{v}^- = \boldsymbol{h}_k - T_k \boldsymbol{x}^{(\nu)} && (2.46)\\ & \boldsymbol{y}_k^{(\nu)} \geq \boldsymbol{0}, \ \boldsymbol{v}^+, \boldsymbol{v}^- \geq \boldsymbol{0}, && (2.47) \end{aligned}$$

where $\boldsymbol{e}^\top = (1,\ldots,1)$ is a vector of ones of the appropriate dimension, which is used to sum the vectors $\boldsymbol{v}^+$ and $\boldsymbol{v}^-$. (The inequality $\widetilde{w} > 0$ means that there is no feasible $\boldsymbol{y} \geq \boldsymbol{0}$.)

If at any stage there is a value k with $\widetilde{w} > 0$, let $\boldsymbol{\sigma}_k^{(v)}$ be the associated vector of Lagrange multipliers (which we obtain by solving the dual of the above LP, (2.45)–(2.47)), and define

$$\boldsymbol{D}_{t+1} = \left(\boldsymbol{\sigma}_k^{(v)}\right)^\top T_k, \tag{2.48}$$

$$d_{t+1} = \left(\boldsymbol{\sigma}_k^{(v)}\right)^\top \boldsymbol{h}_k, \tag{2.49}$$

set $t = t+1$, go back to **Step 1**.

Otherwise, $\widetilde{w} = 0$ for all $k \in \{1,2,\ldots,r\}$, which means that given $\boldsymbol{x}^{(v)}$, for every realisation of $\boldsymbol{\xi}$ the second-stage program has a feasible solution $\boldsymbol{y}_k^{(v)}$.

This in turn implies we can proceed to **Step 3**, which considers the optimality of the solution since everything will now be feasible.

Step 3. Optimality cuts. For $k = 1,\ldots,r$ (consider all realisations of $\boldsymbol{\xi}$), solve the LP

$$\min \quad \widehat{w} = \boldsymbol{q}^\top \boldsymbol{y}_k^{(v)} \tag{2.50}$$

$$\text{such that} \quad W\boldsymbol{y}_k^{(v)} = \boldsymbol{h}_k - T_k\boldsymbol{x}^{(v)} \tag{2.51}$$

$$\boldsymbol{y}_k^{(v)} \geq \boldsymbol{0}. \tag{2.52}$$

Let $\boldsymbol{\pi}_k^{(v)}$ be the vector of Lagrange multipliers associated with an optimal solution for a given k, and define

$$\boldsymbol{E}_{s+1} = \sum_{k=1}^{r} p_k \left(\boldsymbol{\pi}_k^{(v)}\right)^\top T_k, \tag{2.53}$$

$$e_{s+1} = \sum_{k=1}^{r} p_k \left(\boldsymbol{\pi}_k^{(v)}\right)^\top \boldsymbol{h}_k, \tag{2.54}$$

and set

$$J^{(\nu)} = e_{s+1} - \boldsymbol{E}_{s+1}\boldsymbol{x}^{(\nu)}.$$

If $\theta^{(\nu)} \geq J^{(\nu)}$, stop the iteration procedure, as $\boldsymbol{x}^{(\nu)}$ is optimal. Otherwise, set $s = s+1$ and go to **Step 1**.

***Example* 33** (Optimality cuts)**.**

$$\left.\begin{array}{rcrcrcl} \min & z & = & 100x_1 & + & 150x_2 & + \quad \mathbb{E}_{\boldsymbol{\xi}}(q_1y_1 + q_2y_2) \\ \text{such that} & & & x_1 & + & x_2 & \leq \quad 120 \\ & & & 6y_1 & + & 10y_2 & \leq \quad 60x_1 \\ & & & 8y_1 & + & 5y_2 & \leq \quad 80x_2 \\ & & & & & y_1 & \leq \quad d_1 \\ & & & & & y_2 & \leq \quad d_2 \\ & & & & & x_1 & \geq \quad 40 \\ & & & & & x_2 & \geq \quad 20 \end{array}\right\} \tag{2.55}$$

where the random vector $\boldsymbol{\xi} = (d_1, d_2, q_1, q_2)^\top$ has two vector values

$$\begin{aligned} \boldsymbol{\varepsilon}_1 &= (500, 100, -24, -28)^\top \quad \text{with probability } 0.4, \\ \boldsymbol{\varepsilon}_2 &= (300, 300, -28, -32)^\top \quad \text{with probability } 0.6. \end{aligned}$$

We first observe that the second stage is feasible for every $\boldsymbol{x}$ that is feasible to the first-stage problem, because $\boldsymbol{x} > \boldsymbol{0}$ and $\boldsymbol{d} = (d_1, d_2)^\top > \boldsymbol{0}$. (In other words, W is a relatively complete recourse matrix.) We write the first stage LP as

$$\begin{array}{rcrcrcl} \min & z & = & 100x_1 & + & 150x_2 & + \quad \theta \\ & \text{s.t.} & & x_1 & + & x_2 & \leq \quad 120 \\ & & & & & x_1 & \geq \quad 40 \\ & & & & & x_2 & \geq \quad 20, \end{array}$$

where initially we set $\theta = 0$.

The second stage consists of two LPs based on the two realisations of $\boldsymbol{\xi}$ and the constraint

$$W\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\varepsilon}_i) - T(\boldsymbol{\varepsilon}_i)\boldsymbol{x},$$

where

$$W = \begin{pmatrix} 6 & 10 \\ 8 & 5 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \boldsymbol{h}(\boldsymbol{\varepsilon}_1) = \begin{pmatrix} 0 \\ 0 \\ 500 \\ 100 \end{pmatrix}, \quad \boldsymbol{h}(\boldsymbol{\varepsilon}_2) = \begin{pmatrix} 0 \\ 0 \\ 300 \\ 300 \end{pmatrix}$$

$$T(\boldsymbol{\varepsilon}_1) = T(\boldsymbol{\varepsilon}_2) = \begin{pmatrix} -60 & 0 \\ 0 & -80 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{q}(\boldsymbol{\varepsilon}_1) = \begin{pmatrix} -24 \\ -28 \end{pmatrix}, \quad \boldsymbol{q}(\boldsymbol{\varepsilon}_2) = \begin{pmatrix} -28 \\ -32 \end{pmatrix}.$$

Step 0. We initialise by setting

$$t = s = \nu = 0, \quad \boldsymbol{D}_0 = \boldsymbol{0}, \quad d_0 = 0, \quad \boldsymbol{E}_0 = \boldsymbol{0}, \quad e_0 = 0.$$

First Iteration:

Step 1. We set $\nu = \nu + 1 = 1$ and solve

$$\begin{array}{llrcrcl} \min & z & = & 100x_1 & + & 150x_2 & \\ & \text{s.t.} & & x_1 & + & x_2 & \leq 120 \\ & & & & & x_1 & \geq 40 \\ & & & & & x_2 & \geq 20 \end{array}$$

to get $\boldsymbol{x}^{(1)} = (40, 20)^\top$, with $z^{(1)} = 7000$. As there are no constraints (2.43), we set $\theta^{(1)} = -\infty$.

Step 2. We bypass this, as the second-stage is always feasible for every $\boldsymbol{x}$ that is feasible for the first-stage problem.

Step 3. For $\boldsymbol{\varepsilon}_1 = (500, 100, -24, -28)^\top$, we solve

$$\begin{array}{llrcrcl} \min & w & = & -24y_1 & - & 28y_2 & \\ & \text{s.t.} & & 6y_1 & + & 10y_2 & \leq 60x_1 = 2400 \\ & & & 8y_1 & + & 5y_2 & \leq 80x_2 = 1600 \\ & & & & & y_1 & \leq 500 \\ & & & & & y_2 & \leq 100 \\ & & & & & y_1, y_2 & \geq 0, \end{array}$$

to obtain $\boldsymbol{y}_1^{(1)} = (137.5, 100)^\top$ with $w_1^{(1)} = -6100$. The dual LP here yields solution

$$\boldsymbol{\pi}_1^{(1)} = (0, -3, 0, -13)^\top.$$

(Note that the dimensions, bounds and signs for the dual LP variables $\boldsymbol{\pi}$ depend on the form of the primal LP.)

For $\boldsymbol{\varepsilon}_2 = (300, 300, -28, -32)^\top$, we solve

$$\begin{array}{llrcrcl} \min & w & = & -28y_1 & - & 32y_2 & \\ & \text{s.t.} & & 6y_1 & + & 10y_2 & \leq 60x_1 = 2400 \\ & & & 8y_1 & + & 5y_2 & \leq 80x_2 = 1600 \\ & & & & & y_1 & \leq 300 \\ & & & & & y_2 & \leq 300 \\ & & & & & y_1, y_2 & \geq 0, \end{array}$$

to get $\boldsymbol{y}_2^{(1)} = (80, 192)^\top$ with $w_2^{(1)} = -8384$. The dual LP has the solution

$$\boldsymbol{\pi}_2^{(1)} = (-2.32, -1.76, 0, 0)^\top.$$

Now, using $\boldsymbol{h}(\boldsymbol{\varepsilon}_1), \boldsymbol{h}(\boldsymbol{\varepsilon}_2), T(\boldsymbol{\varepsilon}_1)$ and $T(\boldsymbol{\varepsilon}_2)$, we get

$$\begin{aligned} e_1 &= 0.4\boldsymbol{\pi}_1^{(1)}\boldsymbol{h}(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(1)}\boldsymbol{h}(\boldsymbol{\varepsilon}_2) = -520, \\ \boldsymbol{E}_1 &= 0.4\boldsymbol{\pi}_1^{(1)}T(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(1)}T(\boldsymbol{\varepsilon}_2) = (83.52, 180.48)^\top, \\ J^{(1)} &= e_1 - \boldsymbol{E}_1\boldsymbol{x}^{(1)} = -520 - (83.52, 180.48)\boldsymbol{x}^{(1)} = -7470.4. \end{aligned}$$

As $J^{(1)} = -7470.4 > \theta^{(1)} = -\infty$, we add the optimality cut

$$83.52x_1 + 180.48x_2 + \theta \geq -520$$

to the original first stage LP and perform the second iteration.

Second Iteration:

Step 1: We set $\nu = \nu + 1 = 2$ and solve

$$\begin{array}{rrrrrrrrr} \min & z & = & 100x_1 & + & 150x_2 & + & \theta & \\ \text{s.t.} & & & x_1 & + & x_2 & & & \leq & 120 \\ & & & 83.52x_1 & + & 180.48x_2 & + & \theta & \geq & -520 \\ & & & & & & & x_1 & \geq & 40 \\ & & & & & & & x_2 & \geq & 20, \end{array}$$

to get $\boldsymbol{x}^{(2)} = (40, 80)^\top, \theta^{(2)} = -18299.2$, with $z^{(2)} = -2299.2$.

Step 2: Again, since we will always have a feasible $\boldsymbol{y}$ for all first stage solutions $\boldsymbol{x}$, we skip this step.

Step 3:

$$\begin{aligned} \boldsymbol{y}_1^{(2)} &= (400, 0)^\top, w_1^{(2)} = -9600, \text{ with } \boldsymbol{\pi}_1^{(2)} = (-4, 0, 0, 0)^\top \\ \boldsymbol{y}_2^{(2)} &= (300, 60)^\top, w_2^{(2)} = -10320 \text{ with } \boldsymbol{\pi}_2^{(2)} = (-3.2, 0, -8.8, 0)^\top \\ e_2 &= 0.4\boldsymbol{\pi}_1^{(2)} \boldsymbol{h}(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(2)} \boldsymbol{h}(\boldsymbol{\varepsilon}_2) = -1584, \\ \boldsymbol{E}_2 &= 0.4\boldsymbol{\pi}_1^{(2)} T(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(2)} T(\boldsymbol{\varepsilon}_2) = (211.2, 0)^\top, \\ J^{(2)} &= e_2 - \boldsymbol{E}_2 \boldsymbol{x}^{(2)} = -10032 > \theta^{(2)} = -18299.2. \end{aligned}$$

Hence, we add the following optimality cut to the first stage LP

$$211.2x_1 + 0x_2 + \theta \geq -1584.$$

Third Iteration:

Step 1: We set $\nu = \nu + 1 = 3$ and solve

$$\begin{array}{rrrrrrrrr} \min & z & = & 100x_1 & + & 150x_2 & + & \theta & \\ \text{s.t.} & & & x_1 & + & x_2 & & & \leq & 120 \\ & & & 83.52x_1 & + & 180.48x_2 & + & \theta & \geq & -520 \\ & & & 211.2x_1 & & & + & \theta & \geq & -1584 \\ & & & & & & & x_1 & \geq & 40 \\ & & & & & & & x_2 & \geq & 20 \end{array}$$

to get $\boldsymbol{x}^{(3)} = (66.828, 53.172)^\top, \theta^{(3)} = -15697.993$, with $z^{(3)} = -1039.375$.

Step 2: Skipping this step, again.

Step 3.

$$\begin{aligned}
&\boldsymbol{y}_1^{(3)} = (469.224, 100)^\top, w_1^{(3)} = -14061.37, \quad \text{with } \boldsymbol{\pi}_1^{(3)} = (0, -3, 0, -13)^\top,\\
&\boldsymbol{y}_2^{(3)} = (300, 220.966)^\top, w_2^{(3)} = -15470.903, \quad \text{with } \boldsymbol{\pi}_2^{(3)} = (-3.2, 0, -9.8, 0)^\top,\\
&e_3 = 0.4\boldsymbol{\pi}_1^{(3)}\boldsymbol{h}(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(3)}\boldsymbol{h}(\boldsymbol{\varepsilon}_2) = -2104,\\
&\boldsymbol{E}_3 = 0.4\boldsymbol{\pi}_1^{(3)}T(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(3)}T(\boldsymbol{\varepsilon}_2) = (115.2, 96)^\top,\\
&J^{(3)} = e_3 - \boldsymbol{E}_3\boldsymbol{x}^{(3)} = -10032 > \theta^{(3)} = -14907.09.
\end{aligned}$$

Hence, we add the cut

$$115.2x_1 + 96x_2 + \theta \geq -2104.$$

Fourth Iteration:

Step 1. We set $\nu = \nu + 1 = 4$ and solve

$$\begin{array}{rrrrrrrrr}
\min & z & = & 100x_1 & + & 150x_2 & + & \theta & \\
\text{s.t.} & & & x_1 & + & x_2 & & & \leq & 120\\
& & & 83.52x_1 & + & 180.48x_2 & + & \theta & \geq & -520\\
& & & 211.2x_1 & & & + & \theta & \geq & -1584\\
& & & 115.2x_1 & + & 96x_2 & + & \theta & \geq & -2104\\
& & & & & & & x_1 & \geq & 40\\
& & & & & & & x_2 & \geq & 20
\end{array}$$

to get $\boldsymbol{x}^{(4)} = (40, 33.75)^\top, \theta^{(4)} = -9952$, with $z^{(4)} = -889.5$.

Step 2: We are skipping this step, again!

Step 3.

$$\begin{aligned}
&\boldsymbol{y}_1^{(4)} = (300,60)^\top,\ w_1^{(4)} = -8880, \quad \text{with } \boldsymbol{\pi}_1^{(4)} = (-2.08,-1.44,0,0)^\top,\\
&\boldsymbol{y}_2^{(4)} = (300,60)^\top,\ w_2^{(4)} = -10320 \quad \text{with } \boldsymbol{\pi}_2^{(4)} = (-3.175,-0.05,-8.548,0)^\top,\\
&e_4 = 0.4\boldsymbol{\pi}_1^{(4)}\boldsymbol{h}(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(4)}\boldsymbol{h}(\boldsymbol{\varepsilon}_2) = -1538.601,\\
&\boldsymbol{E}_4 = 0.4\boldsymbol{\pi}_1^{(4)}T(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(4)}T(\boldsymbol{\varepsilon}_2) = (164.212,48.501)^\top,\\
&J^{(4)} = e_4 - \boldsymbol{E}_4\boldsymbol{x}^{(4)} = -9744 > \theta^{(4)} = -9952
\end{aligned}$$

Hence, we add the cut

$$164.212x_1 + 48.501x_2 + \theta \geq -1538.601.$$

Fifth Iteration:

Step 1. We set $\nu = \nu + 1 = 5$ and solve

$$\begin{array}{lrrcrcrcr}
\min & z & = & 100x_1 & + & 150x_2 & + & \theta & \\
\text{s.t.} & & & x_1 & + & x_2 & & & \leq & 120\\
& & & 83.52x_1 & + & 180.48x_2 & + & \theta & \geq & -520\\
& & & 211.2x_1 & & & + & \theta & \geq & -1584\\
& & & 115.2x_1 & + & 96x_2 & + & \theta & \geq & -2104\\
& & & 164.212x_1 & + & 48.501x_2 & + & \theta & \geq & -1538.601\\
& & & & & & & x_1 & \geq & 40\\
& & & & & & & x_2 & \geq & 20
\end{array}$$

to get $\boldsymbol{x}^{(5)} = (46.667,36.25)^\top, \theta^{(5)} = -10960$, with $z^{(5)} = -855.833$.

Skipping **Step 2** again, we perform **Step 3**:

$$
\begin{aligned}
\boldsymbol{y}_1^{(5)} &= (300,100)^\top,\ w_1^{(5)} = -10000 \quad \text{with} \\
\boldsymbol{\pi}_1^{(5)} &= (-0.695,-2.479,0,-8.655)^\top \\
\boldsymbol{y}_2^{(5)} &= (300,100)^\top,\ w_2^{(5)} = -11600 \quad \text{with} \\
\boldsymbol{\pi}_2^{(5)} &= (-3.156,-0.088,-8.362,0)^\top \\
e_5 &= 0.4\boldsymbol{\pi}_1^{(5)}\boldsymbol{h}(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(5)}\boldsymbol{h}(\boldsymbol{\varepsilon}_2) = -1851.361 \\
\boldsymbol{E}_5 &= 0.4\boldsymbol{\pi}_1^{(5)}T(\boldsymbol{\varepsilon}_1) + 0.6\boldsymbol{\pi}_2^{(5)}T(\boldsymbol{\varepsilon}_2) = (130.307,83.522)^\top \\
\left(J^{(5)} = e_5 - \boldsymbol{E}_5\boldsymbol{x}^{(5)} = -10960\right) &= \left(\theta^{(5)} = -10960\right)
\end{aligned}
$$

Stop as $\boldsymbol{x}^{(5)} = (46.667,36.25)^\top$, with $z^{(5)} = -855.833$ is optimal.

Chapter 3

Markov Decision Chains

This section follows **very, very closely** the lecture notes of Richard Weber at the University of Cambridge. Very good reference books are [**?**] and [**?**] (also, followed **very** closely in parts).

We are now going to be concerned with *dynamical systems* and their optimisation *over time*. We will focus on stochastic systems, which are therefore dynamic and which are related to the *stochastic linear programs* we have seen (multi-stage recourse problems).

Markov decision processes are essentially models for sequential decision making when outcomes are uncertain. As we will see in this course, Markov decision process models consist of *decision epochs*, *states*, *actions*, *rewards/costs* and *transition probabilities*. Choosing an action in a state at a decision epoch generates a reward/cost, and determines the state at the next decision epoch through a transition probability function.

We will encounter the concept of *policies* or *strategies* which prescribe particular actions to choose under any eventuality at every future decision epoch. As decision makers, we seek policies that are optimal in some sense.

People make decisions every day and those decisions have both short term and long term consequences. The same is true for the actions taken within these Markov decision processes and therefore, we should take care

that decisions are not made in isolation, since any decision taken now impacts any subsequent decision, which impacts subsequent decisions...

Defn 3.1. ***A sequential decision model*** *is as follows. At a specific point in time a decision maker observes the state of a system. Based on this state, the decision maker chooses an action. This action results in two outcomes:*

1. *an immediate reward or cost and*
2. *the evolution of the process to a new state at a subsequent point in time according as some probability distribution.*

At the subsequent point in time the decision maker faces a similar problem, but maybe with the system in a different state and with a different set of possible actions.

We will assume that we have a random process which can, to some extent, be controlled. For example, at certain decision points, we have a choice of actions that can be taken, where different actions have the potential to affect the future behaviour of the process in different ways.

When actions can be taken to influence the evolution of a Markov process, then we call this a **Markov decision process**, or alternatively, a **Markov decision problem**. Typically, the aim of exerting some influence on the process is to minimise or maximise a reward or cost function, which in turn depends on the state trajectory of the Markov process.

§3.1 MOTIVATION

***Example* 34** (Machine repair I: A cost minimisation problem)**.** Consider the problem of managing the operation of a machine efficiently over T time

periods, corresponding to $T+1$ stages, labelled $0, 1, \ldots, T$. During each time period, the machine can be in any one of N states, denoted $1, 2, \ldots, N$:

- State 1 corresponds to the machine being in perfect condition.

  $\vdots$

- State N corresponds to the machine being in greatest disrepair.

Let $h(i)$ be the operating cost of the machine during one time period when the machine is in state i, and assume that

$$h(1) \leq h(2) \leq h(3) \leq \ldots \leq h(N).$$

During each of the T time periods, in the absence of repair, the state of the machine can either remain the same or deteriorate to one of the higher states of disrepair. Let $p_{ij} = P(\text{next state is } j | \text{current state is } i)$, with $p_{ij} = 0$ if $j < i$, for $i, j = 1, \ldots, N$.

At each stage of operation (the start of each time period), we know the state of the machine, and we must choose one of the following possible actions:

1. First repair the machine to perfect working order (state 1) at a cost R immediately prior to the operating period.

2. Let the machine operate one more time period in its current state.

Assume that once repaired, the machine is guaranteed to stay in state 1 until the next stage (until the beginning of the next time period). In subsequent periods, it may deteriorate to states $j > 1$ according to the transition probabilities p_{1j}. At the Tth stage (the end of the Tth time period), the operation of the machine stops, and the cost incurred is zero for all states of the machine.

The objective is, *to determine, for each time period and for each state of the machine, whether the optimal action is to repair, or leave the machine in its current state.* In particular, we wish to obtain a set of instructions, called a

policy, which prescribes the sequence of actions to take when the machine starts in a given state i_0, such that the total expected cost of operation over the T time periods is minimised.

Consider a specific instance of the machine repair problem over $T = 4$ time periods, and $N = 3$ states. Let the operating cost in each state and cost of repair be given by $h(1) = 2, h(2) = 4, h(3) = 6, R = 6$. The decision stages are $n = 0, 1, 2, 3, 4$.

If the action REPAIR is chosen, the state transition probabilities are given by

$$P(\text{repair}) = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

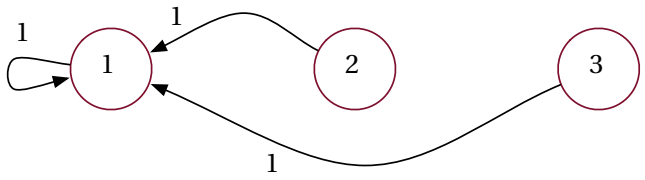


Figure 3.1: *State transition diagram if action* REPAIR *is chosen.*

Alternatively, if the action DON'T REPAIR is chosen, the state transition probabilities are given by

$$P(\text{don't repair}) = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0 & 0.6 & 0.4 \\ 0 & 0 & 1 \end{pmatrix}.$$

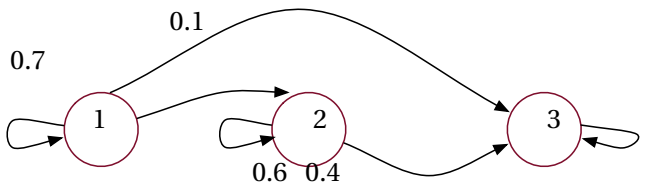


Figure 3.2: *State transition diagram if the action* DON'T REPAIR *is chosen.*

The transition probabilities of the Markov chain at stage (time) n therefore depend on the action that is chosen at stage n. Let X_n represent the state of the machine at stage n, that is, at the beginning of the nth operation period. If we were to employ the same action at each decision stage, then the Markov process $X_n, n = 0,\ldots,4$, on the states $\{1,2,3\}$, would be time-homogeneous. On the other hand, if we were to alternate the action at each stage, then the Markov process would be time-inhomogeneous.

This machine repair example is a *finite horizon problem*, because T is finite. Furthermore, T is a deterministic quantity which is given in advance.

§3.2 ELEMENTS OF A MARKOV DECISION CHAIN

We now focus on stochastic systems, in particular where the evolution of the process is governed by a DTMC. A **Markov decision chain** (MDC) $\{X_n\}_{n\in\mathbb{N}}$ consists of:

1. **States**, in a **state space** $\mathscr{S} \subseteq \mathbb{N}$. For example, $\mathscr{S} = \{0,1,\ldots,N\}$. $X_n \in \mathscr{S}$ for all $n = 0,1,\ldots$. We will often assume $\mathscr{S} \subset \mathbb{N}$.

2. **Actions**, $u_n \in \mathscr{U}_n(x_n)$, for all $n = 0,1\ldots$. We will often have $\mathscr{U}(x_n)$, where the set of actions available depends upon the current state of the chain only.

3. **Costs**, $c(x_n,u_n,n)$. Assumed separable (decomposable) — only incurred at current stage and dependent upon action. We will often assume $c(x_n,u_n)$, that is, the cost depends only upon current state and/or action, and not on time.

   We also have a set of **terminal costs** $C(x_T)$, which are the costs incurred in the final stage T (if $T < \infty$) and are dependent on the state x_T. This is often assumed to be 0, but not always.

4. **Transition probability distributions**. Let

$$\mathbf{x}_n = (x_1, x_2, \ldots, x_n) \quad \text{and} \quad \mathbf{u}_n = (u_0, u_1, \ldots, u_n)$$

denote the state and action histories at time n, respectively. We must specify

$$\begin{aligned}\Pr(X_{n+1} = x_{n+1}|\mathbf{x}_n, \mathbf{u}_n, n) &=^* \Pr(X_{n+1} = x_{n+1}|x_n, u_n, n)\\ &= p(x_{n+1}|x_n, u_n),\end{aligned}$$

where $=^*$ is a consequence of Markov property.

5. **Optimisation criterion**:

   a) **Finite horizon.** In this case, the behaviour of the system is considered for epochs $n = 0$ to $n = T$, for a fixed positive integer T.

   b) **Infinite horizon.** In this case, $n = 0, 1, \ldots$. We will consider this situation with and without *discounting*. *Average cost* also rests in this category.

POLICIES

A **policy** (or **strategy**) is a rule for choosing the action under all possible circumstances as a function of the circumstances. A controller's actions under a policy can be based on the previous states visited, the actions chosen in those states, and when those visits occurred.

There are several important types of policies, these are classified by how much of the history may be utilised by the controller:

- A **stationary policy**, denoted by f, operates as follows: Associated with each state i is a distinguished action $u(i) \in \mathcal{U}_i$. If at any time the *controller* finds the system in state i, then the controller always chooses the action $u(i)$.

  Thus, a stationary policy depends on the history of the process only through the current state. To implement a stationary policy, the controller needs to know the current state of the system only.

- A slightly less restrictive type of policy is the **randomised stationary policy**, denoted by f_R. Associated with each state i is a pmf $P^i_{f_R}$ on $\mathscr{U}_i$. If at any time the controller finds the system in state i, then the controller always chooses action u with prob. $P^i_{f_R}(u)$.

  As is the case for a stationary policy, a randomised stationary policy depends on the history of the process only through the current state. A stationary policy can be viewed as a degenerate randomised stationary policy.

- A **deterministic Markov policy** is a sequence $\theta = (f_0, f_1, f_2, \ldots)$ of stationary policies. It operates as follows: If the chain is in state i at time $t = n$, then the controller chooses action $f_n(i)$. Thus a deterministic Markov policy depends on the history of the chain only through the current state and the time index.

- A **randomised Markov policy** is a sequence $\theta = (f_{R0}, f_{R_1}, f_{R_2}, \ldots)$ of randomised stationary policies. It operates as follows: It the chain is in state i at time $t = n$, then the controller chooses action $u \in \mathscr{U}_i$ with probability $P^i_{R_n}(u)$. Thus a randomised Markov policy depends on the history of the chain only through the current state and the time index.

§3.3 FINITE HORIZON PROBLEMS

Defn 3.2. *A **basic finite horizon problem** is a discrete-time Markov chain with finite time horizon T, where T is given in advance. It takes values on a finite state space $\mathscr{S} = \{1, \dots N\}$.*

For each state $i \in \mathscr{S}$ and each $n \in \{0, 1, \dots T-1\}$, there is a finite set of actions $\mathscr{U}_n(i)$ that can be selected. No action needs to be selected at the terminating time T.

Transitions between states (i, j) are governed by the probabilities $p_{ij}(u_n)$, where $u_n \in \mathscr{U}_n(i)$. So the state transition probabilities in state i depend on the particular action $u_n \in \mathscr{U}_n(i)$ that is selected. We assume that the transition probabilities are a function of n only through the actions u_n.

Let $c(i, u_n)$ be the expected single-stage cost incurred by selecting action u_n when in state i at time n, which is the expected cost incurred during the time period n to $n+1$, when the process is in state i at time n and the action u_n is selected.

If we assume that the expected single stage costs $c(i, u_n)$ are stationary, then this means that the cost $c(i, u_n)$ depends on n only through the action u_n.

*Let $C(i), i \in \mathscr{S}$, be the **terminal cost** associated with ending in state i; often this is set to zero.*

We will consider a class of **policies** Ψ, each consisting of a sequence of functions

$$\Psi = \{f_0, f_1, \dots, f_{T-1}\},$$

where each function f_n maps states $i \in \mathscr{S}$ into actions:

$$u_n = f_n(i).$$

We assume that $f_n(i) \in \mathscr{U}_n(i)$ for all $i \in \mathscr{S}$; such a policy is called **admissible**. Without this constraint, the policy does not make sense.

Given a policy Ψ, the state evolution is a Markov process, X_n^Ψ, on the state space $\mathscr{S}$, for $n = 0, \ldots, T$, with transition probabilities

$$P\left(X_{n+1}^\Psi = j \mid X_n^\Psi = i\right) = p_{ij}\left(f_n(i)\right) \quad \text{for } i, j \in \mathscr{S} \text{ and } n = 0, \ldots, T-1.$$

The expected cost of a policy Ψ, starting from state i at the beginning of stage $k \in \{0, 1, \ldots, T\}$, is defined as

$$J_k^\Psi(i) = \mathbb{E}\left[C(X_T^\Psi) + \sum_{n=k}^{T-1} c(X_n^\Psi, f_n(X_n^\Psi))\right], \quad \text{with } X_k^\Psi = i. \tag{3.1}$$

The objective in solving the finite horizon MDP is to obtain a policy Ψ that minimises the expected cost for $k = 0$

$$J_0^\Psi(i_0) = \mathbb{E}\left[C(X_T^\Psi) + \sum_{n=0}^{T-1} c(X_n^\Psi, f_n(X_n^\Psi))\right], \tag{3.2}$$

for a given starting state $X_0^\Psi = i_0$. That is, we wish to minimise the expected total cost over all stages (recall that decisions are only made at stages $0, 1, \ldots T-1$, and none at stage T).

Defn 3.3. *For a given initial state i_0, an **optimal policy***

$$\Psi^* = \{f_0^*, f_1^*, \ldots, f_{T-1}^*\}$$

is a policy that satisfies

$$J^{\Psi^*}(i_0) = \min_{\Psi \in \Gamma} J_0^\Psi(i_0),$$

where Γ is the set of all admissible policies.

The **optimal cost** depends on the starting state i_0, and is denoted $J^*(i_0)$:

$$J^*(i_0) = J^{\Psi^*}(i_0) = \min_{\Psi \in \Gamma} J_0^{\Psi}(i_0).$$

We may view J^* as a function that assigns to each initial state $i_0 \in \mathscr{S}$ the optimal cost $J^*(i_0)$.

Finite horizon problems can be solved using the technique known as **dynamic programming**, which rests on the Principle of Optimality, which is stated as follows.

Theorem 3.1 (Principle of Optimality)**.** *Let*

$$\Psi^* = \{f_0^*, f_1^*, \ldots, f_{T-1}^*\}$$

be an optimal policy starting from a given state i_0. Assume that when using Ψ^*, the system visits state i at stage k with positive probability; that is,

$$P(X_k^{\Psi^*} = i \mid X_0 = i_0) > 0.$$

Consider the sub-problem in which the process is in state i at stage k, $X_k^{\Psi^*} = i$, and we wish to minimise the expected cost incurred over the "time-to-go" or remaining $T-k$ stages, given by

$$\mathbb{E}\left[C(X_T^{\Psi}) + \sum_{n=k}^{T-1} c(X_n^{\Psi}, f_n(X_n^{\Psi}))\right].$$

Then, the truncated policy $\{f_k^*, f_{k+1}^*, \ldots, f_{T-1}^*\}$ is optimal for this sub-problem.

A sketch of the proof is, if the truncated policy were not optimal, then the entire policy Ψ^* would not be optimal. An analogy: Suppose that the shortest route to travel between Adelaide and Paris is via Frankfurt. It follows that the

part of that route between Frankfurt and Paris is also the shortest route for the "sub-problem" of travelling from Frankfurt to Paris in minimum time.

The Principle of Optimality suggests that an optimal policy can be constructed piece-by-piece, starting from the final stage of the decision process and working backwards, until an optimal policy for the entire problem is constructed.

This piece-by-piece technique is precisely what is done by the **dynamic programming algorithm.** We introduce the dynamic programming algorithm by applying it to the machine repair problem in Example 34, which is a basic finite horizon problem and so fits within this general scheme.

***Example* 35** (Machine repair II)**.** In the machine repair problem, the terminal costs are $C(1) = C(2) = C(3) = 0$ and for every state $i = 1,2,3$, and all stages $n = 0,\ldots,4$, we have two actions

$$\mathscr{U}_n(i) = \{\text{REPAIR}, \text{DON'T REPAIR}\} = \{\text{R}, \text{D}\},$$

at each decision stage. For each pair of states (i, j), we have the transition probabilities $p_{ij}(1)$ and $p_{ij}(2)$ corresponding to actions 1 and 2, respectively. The complete set of transition probabilities are contained in the matrices

$$P(\text{R}) = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \text{ and } \quad P(\text{DON'T REPAIR}) = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0 & 0.6 & 0.4 \\ 0 & 0 & 1 \end{pmatrix}.$$

The dynamic programming algorithm proceeds as follows: we work out costs by starting with the last time step and working backwards.

Stage T = 4: No decision needs to be made because this is the end of the operation, and so $J_4(i) = 0$, for $i = 1,2,3$.

Stage T − 1 = 3: At the beginning of stage 3, the machine can be in any of the states 1,2 or 3, depending on previous decisions.

- Suppose at the beginning of stage 3, the machine is in state 1:

$$u \in \{\text{REPAIR}, \text{DON'T REPAIR}\} = \{\text{R}, \text{D}\}.$$

Because the process terminates at time 4, we consider the expected single-stage cost only. So the single-stage cost that takes us from stage 3 to 4 is the only cost that needs to be considered for the first iteration of the dynamic programming algorithm.

Hence[1],

$$\begin{aligned} J_3(1) &= \min_{u \in \{\text{R},\text{D}\}} \mathbb{E}[c(1,u)] = \min\{\mathbb{E}[c(1,\text{R})], \mathbb{E}[c(1,\text{D})]\} \\ &= \min\left\{R + h(1), p_{11}(\text{D})h(1) + p_{12}(\text{D})h(2) + p_{13}(\text{D})h(3)\right\} \\ &= \min[6+2, 0.7\times 2 + 0.2\times 4 + 0.1\times 6] \\ &= \min[6+2, 2.8] \\ &= 2.8. \end{aligned}$$

This tells us that if the machine is in state 1 at time 3, then the optimal action is not to repair the machine. Taking this action minimises the expected remaining cost (to reach the termination stage), which is equal to 2.8. This optimal cost is sometimes refereed to as the **cost-to-go**.

We now have to repeat this calculation for the other states.

- Suppose the machine is in state 2 at time 3 (we don't care how it got there, for now). Here, the dynamic programming algorithm requires us

[1] Note: we assume that the transition from state i to state j happens immediately once an action is made, rather than over a unit of time. So If we are in state i and time n and take an action, then we immediately transition into state j according to probability p_{ij}(said action), and we will be paying the maintenance cost for being in state j over time interval $[n, n+1]$.

to set

$$\begin{aligned} J_3(2) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}[c(2,u)] = \min\{\mathbb{E}[c(2,\mathrm{R})],\mathbb{E}[c(2,\mathrm{D})]\} \\ &= \min\left\{R + h(1), p_{21}(\mathrm{D})h(1) + p_{22}(\mathrm{D})h(2) + p_{23}(\mathrm{D})h(3)\right\} \\ &= \min\{6+2, 0.6\times 4 + 0.4\times 6\} \\ &= \min\{8, 4.8\} = 4.8. \end{aligned}$$

The action $u = \mathrm{D}$ corresponding to DON'T REPAIR, is thus the action which should be selected if the machine is in state 2 at time 3, which incurs a cost-to-go of 4.8.

- Similarly,

$$\begin{aligned} J_3(3) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}[c(3,u)] = \min\{\mathbb{E}[c(3,\mathrm{R})],\mathbb{E}[c(3,\mathrm{D})]\} \\ &= \min\left\{R + h(1), p_{31}(\mathrm{D})h(1) + p_{32}(\mathrm{D})h(2) + p_{33}(\mathrm{D})h(3)\right\} \\ &= \min[6+2, 6] \\ &= 6; \end{aligned}$$

thus the optimal action is DON'T REPAIR, which incurs a cost-to-go of 6.

We have therefore identified the optimal actions and their associated optimal (minimum) costs for all states that the machine may find itself in at time 3. That is, we have identified the optimal costs

$$J_3(1) = 2.8, \quad J_3(2) = 4.8, \quad J_3(3) = 6,$$

which result from taking the optimal actions at stage 3.

Stage T − 2 = 2: At the beginning of stage 2, the machine can also be in any of the states 1, 2 or 3, depending on the previous decision (we don't care how it got there, for now).

- Suppose the machine is in state 1. Since there are two stages to go until termination, we now wish to select the action that minimises

the expected cost over both stages, rather than just minimising the expected single-stage cost from stage 2 to 3.

The quantity that we therefore need to minimise is

$$\mathbb{E}[\text{single-stage cost from stage 2 to 3}] + \mathbb{E}[\text{cost from stage 3 to 4} \mid \text{the optimal action is taken at stage 3}].$$

The dynamic programming algorithm requires us to set

$$\begin{aligned} J_2(1) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}\left[c(1,u) + \sum_{j=1}^{3} p_{1j}(u) J_3(j)\right] \\ &= \min\left\{R + h(1) + \sum_{j=1}^{3} p_{1j}(\mathrm{R}) J_3(j), \quad p_{11}(\mathrm{D})h(1) + p_{12}(\mathrm{D})h(2) + p_{13}(\mathrm{D})h(3) + \sum_{j=1}^{3} p_{1j}(\mathrm{D}) J_3(j)\right\} \\ &= \min\{6 + 2 + 1 \times 2.8, \qquad (0.7 \times 2 + 0.2 \times 4 + 0.1 \times 6) + (0.7 \times 2.8 + 0.2 \times 4.8 + 0.1 \times 6)\} \\ &= \min\{10.8, \quad 5.78\} = 5.78, \end{aligned}$$

which is obtained by selecting the action DON'T REPAIR.

- Suppose the machine is in state 2 at time 2, then

$$\begin{aligned} J_2(2) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}\left[c(2,u) + \sum_{j=1}^{3} p_{2,j}(u) J_3(j)\right] \\ &= \min\left\{R + h(1) + \sum_{j=1}^{3} p_{2j}(\mathrm{R}) J_3(j), \quad p_{21}(\mathrm{D})h(1) + p_{22}(\mathrm{D})h(2) + p_{23}(\mathrm{D})h(3) + \sum_{j=1}^{3} p_{2j}(\mathrm{D}) J_3(j)\right\} \\ &= \min\{10, \ (0.6 \times 4 + 0.4 \times 6) + (0.6 \times 4.8 + 0.4 \times 6)\} = \min\{10, \ 10.08\} = 10, \end{aligned}$$

which is obtained by selecting the action REPAIR.

- Similarly, suppose the machine is in state 3 at time 2, then

$$\begin{aligned} J_2(3) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}\left[c(3,u) + \sum_{j=1}^{3} p_{3j}(u) J_3(j)\right] \\ &= \min\{10, \quad h(3) + J_3(3)\} = \min\{10, \ 6 + 6\} = 10, \end{aligned}$$

so that in this case, the optimal action is also REPAIR.

We have now identified the optimal costs

$$J_2(1) = 5.78, \quad J_2(2) = 10, \quad J_2(3) = 10,$$

which result from taking the optimal actions at stages 2 and 3.

Stage T − 3 = 1: At time 1, there are three stages to go until termination, and we now select the action that minimises the expected cost over all three stages. At the beginning of stage 1, the machine can again be in any of the states 1, 2 or 3.

- Suppose the machine is in state 1. The dynamic programming algorithm requires us to set

$$\begin{aligned} J_1(1) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}\left[c(1,u) + \sum_{j=1}^{3} p_{1,j}(u) J_2(j)\right] \\ &= \min\left\{R + h(1) + \sum_{j=1}^{3} p_{1j}(\mathrm{R}) J_2(j), \quad p_{11}(\mathrm{D})h(1) + p_{12}(\mathrm{D})h(2) + p_{13}(\mathrm{D})h(3) + \sum_{j=1}^{3} p_{1j}(\mathrm{D}) J_2(j)\right\} \\ &= \min\{6 + 2 + 1 \times 5.78, \qquad (0.7 \times 2 + 0.2 \times 4 + 0.1 \times 6) + (0.7 \times 5.78 + 0.2 \times 10 + 0.1 \times 10)\} \\ &= \min\{13.78, \quad 9.846\} = 9.8460. \end{aligned}$$

  Hence, the optimal action is DON'T REPAIR, with associated cost-to-go of 9.8460.

- Similarly,

$$\begin{aligned} J_1(2) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}\left[c(2,u) + \sum_{j=1}^{3} p_{2,j}(u) J_2(j)\right] \\ &= \min\left\{R + h(1) + \sum_{j=1}^{3} p_{2j}(\mathrm{R}) J_2(j), \quad p_{21}(\mathrm{D})h(1) + p_{22}(\mathrm{D})h(2) + p_{23}(\mathrm{D})h(3) + \sum_{j=1}^{3} p_{2j}(\mathrm{D}) J_2(j)\right\} \\ &= \min\{6 + 2 + 1 \times 5.78, \ \ (0.6 \times 4 + 0.4 \times 6) + (0.6 \times 10 + 0.4 \times 10)\} \\ &= \min\{13.78, \ \ 14.8\} = 13.78, \end{aligned}$$

and

$$\begin{aligned} J_1(3) &= \min_{u\in\{\text{R,D}\}} \mathbb{E}\left[c(3,u) + \sum_{j=1}^{3} p_{3j}(u)J_2(j)\right] \\ &= \min\{13.78, \quad h(3) + J_2(3)\} \\ &= \min\{13.78, \ 6+10\} = 13.78, \end{aligned}$$

which are achieved by selecting REPAIR, for both states 2 and 3.

Stage T − 4 = 0: We are now in a position to calculate the optimal policy and corresponding costs $J_0(i)$ for all possible initial states i of the machine.

The dynamic programming algorithm requires that

$$\begin{aligned} J_0(1) &= \min_{u\in\{\text{R,D}\}} \mathbb{E}\left[c(1,u) + \sum_{j=1}^{3} p_{1j}(u)J_1(j)\right] \\ &= \min\left\{R + h(1) + \sum_{j=1}^{3} p_{1j}(\text{R})J_1(j), \quad p_{11}(\text{D})h(1) + p_{12}(\text{D})h(2) + p_{13}(\text{D})h(3) + \sum_{j=1}^{3} p_{1j}(\text{D})J_1(j)\right\} \\ &= \min\{6+2+1\times 9.846, \qquad (0.7\times 2 + 0.2\times 4 + 0.1\times 6) + (0.7\times 9.846 + 0.2\times 13.78 + 0.1\times 13.78)\} \\ &= \min\{17.846, \quad 13.8262\} = 13.8262. \end{aligned}$$

Similarly,

$$\begin{aligned} J_0(2) &= \min_{u\in\{\text{R,D}\}} \mathbb{E}\left[c(2,u) + \sum_{j=1}^{3} p_{2,j}(u)J_1(j)\right] \\ &= \min\left\{R + h(1) + \sum_{j=1}^{3} p_{2j}(\text{R})J_1(j), \quad p_{21}(\text{D})h(1) + p_{22}(\text{D})h(2) + p_{23}(\text{D})h(3) + \sum_{j=1}^{3} p_{2j}(\text{D})J_1(j)\right\} \\ &= \min\{6+2+1\times 9.846, \ (0.6\times 4 + 0.4\times 6) + (0.6\times 13.78 + 0.4\times 13.78)\} \\ &= \min\{17.846, \ 18.58\} = 17.8460, \end{aligned}$$

and

$$\begin{aligned} J_0(3) &= \min_{u\in\{\mathrm{R},\mathrm{D}\}} \mathbb{E}\left[c(3,u) + \sum_{j=1}^{3} p_{3j}(u) J_1(j)\right] \\ &= \min\{17.846, \quad h(3) + J_1(3)\} \\ &= \min\{17.846, \ 6 + 13.78\} = 17.846, \end{aligned}$$

So, we find that

$$J_0(1) = 13.8262, \quad J_0(2) = 17.846, \quad J_0(3) = 17.846,$$

which are achieved by selecting the action DON'T REPAIR if the machine starts in state 1, and selecting the action REPAIR if the machine starts in either state 2 or 3.

The above algorithm has allowed us to identify an optimal policy

$$\Psi^* = \{f_0^*, f_1^*, f_2^*, f_3^*\},$$

where the functions $f_0^*, \ldots, f_3^*$ are given in the Table 3.1, summarising optimal actions and costs for every possible combination of stage and machine state:

i	$J_0(i)$	$f_0^*(i)$	$J_1(i)$	$f_1^*(i)$	$J_2(i)$	$f_2^*(i)$	$J_3(i)$	$f_3^*(i)$	$J_4(i)$
1	13.8262	don't repair	9.846	don't repair	5.78	don't repair	2.8	don't repair	0
2	17.846	repair	13.78	repair	10	don't repair	4.8	don't repair	0
3	17.846	repair	13.78	repair	10	repair	6.00	don't repair	0

Table 3.1: Optimal Policy

Suppose you had to translate the optimal policy Ψ^* into words, for a manager who is not mathematically inclined. What would you say?

- Don't repair the machine if it is in perfect working order (state 1).

- If the machine is slightly damaged (state 2), only repair the machine if the **time-to-go** is greater than 2.
- If the machine is badly damaged (state 3), only repair the machine if the **time-to-go** is greater than 1.

The above policy makes sense, because as a general rule, if there is not much operation time remaining, the expected cost of investing in machine repair is greater than the expected cost of just letting the machine continue in a damaged state.

Of course, the optimal policy and costs are usually different if we change the problem parameters, such as lowering the cost of fixing the machine from $R = 6$ to $R = 1$. The optimal policy is then to repair the machine whenever it is not in perfect working condition (state 1). (Check!)

Theorem 3.2 (Bellman Optimisation Principle)**.** *For every initial state i_0, the optimal cost $J^*(i_0)$ is equal to $J_0(i_0)$, where the function J_0 is given by the last step of the following algorithm, which proceeds backward in time from stage $T-1$ to stage 0:*

$$J_T(i) = C(i),$$

and then for $k = 0, \ldots, T-1$,

$$J_k(i) = \min_{u_k \in \mathscr{U}_k(i)} \mathbb{E}\left[c(i,u_k) + \sum_{j \in \mathscr{S}} p_{ij}(u_k) J_{k+1}(j) \right], \tag{3.3}$$

where $C(i)$ is the terminal cost associated with state i, for $i \in \mathscr{S}$.

Furthermore, if $u_k^* = f_k^*(i)$ minimises the RHS of (3.3) for each stage $k = 0, \ldots, T-1$ and each state $i \in \mathscr{S}$, then the policy

$$\Psi^* = \{f_0^*, f_1^*, \ldots, f_{T-1}^*\} \quad \textit{is optimal.}$$

Theorem 3.2 is a consequence of the Markov property and is the basis of dynamic computing. The optimality equation (3.3) is a backward recursion in time, such that the later policy is decided first. This process of backward recursion calculating expected cost (reward), is also known as **Value Iteration**.

Let us now look at a manufacturing process where we will maximise rewards r over time. Of course we could minimise the costs c (negative of the rewards) and use the same notation as before, but it is more convenient to scribe it in terms of a maximisation problem.

***Example* 36** (Designer furniture: A value maximisation problem)**.** A small company makes reproduction designer furniture and at the end of the year, the business is in one of two states

$$\{\text{THRIVING}, \text{FLOUNDERING}\} = \{1,2\}.$$

At the beginning of each year, the company can make a decision as to whether they will invest money on design work or not.

If the company decides not to do design work, then the state transitions from the beginning of the year to the end of that same year form a Markov chain with transition matrix on $\{1,2\}$

$$P(\text{NO DESIGN}) = \begin{pmatrix} 0.5 & 0.5 \\ 0.3 & 0.7 \end{pmatrix}.$$

There is also a reward matrix that is associated with the decision NO DESIGN. If the company continues to THRIVE, sales provide a net income of 9. If it moves from THRIVING to FLOUNDERING, the reduced sales during the year provide a net income of only 3. If the business moves from FLOUNDERING to THRIVING, the sales provide an income of 3. If the business remains in the floundering state, the sales revenue does not cover the costs of operation, resulting in a reward of -4.

The resultant reward matrix under NO DESIGN is thus given by

$$R(\text{NO DESIGN}) = \begin{pmatrix} 9 & 3 \\ 3 & -4 \end{pmatrix}.$$

Notice that the rewards take no account of future earnings, as they are realised only over the current stage (year) and the transition probabilities are independent of the past history of the process.

It is possible to increase the chances of moving to (or staying in) the thriving state by spending time on design, which has an associated cost which reduces the rewards by 4. Hence, under DESIGN we get

$$P(\text{DESIGN}) = \begin{pmatrix} 0.75 & 0.25 \\ 0.7 & 0.3 \end{pmatrix},$$

$$R(\text{DESIGN}) = R(\text{NO DESIGN}) - \begin{pmatrix} 4 & 4 \\ 4 & 4 \end{pmatrix} = \begin{pmatrix} 5 & -1 \\ -1 & -8 \end{pmatrix}.$$

We label the two actions NO DESIGN and DESIGN that can be taken at the beginning on the year (stage) for convenience as NO and DES respectively, so that the set of actions are

$$U_n(i) = \{\text{NO}, \text{DES}\} \quad \text{for each state } i = 1, 2.$$

Suppose now that the lease for the premises of the company runs out at the end of $T = 4$ years and that the company members have agreed to split up and go their separate ways. Hence, they must decide whether to spend time and money on design and their decision will be based on expected returns (rewards) or **expected monetary value (EMV)**.

In this example, there is no fixed reward for starting in any particular state at the beginning of a stage, because the reward depends on what state the company is in at the beginning and what state it is in at the end of that stage.

Let $v(i, u_k)$ be the single stage expected reward given that the company starts in state i, at the beginning of stage k under decision (action) u_k. Also, let $p_{ij}(u_k)$ and $r_{ij}(u_k)$ be the corresponding entries in the transition probability matrix and reward matrix, respectively, under decision u_k.

The EMV $V_k^{\Psi}(i)$ of a policy Ψ starting from state $i \in \mathscr{S}$ at the beginning of stage k is defined by a **Bellman equation** given by

$$V_k^{\Psi}(i) = \mathbb{E}\left[V_T\left(X_T^{\Psi}\right) + \sum_{n=k}^{T-1} v\left(X_n^{\Psi}, f(X_n^{\Psi})\right) \right],$$

for $k = 0, 1, \ldots, T-1$ and where $X_k^\Psi = i$ and $V_T\left(X_T^\Psi\right)$ is the value received for finishing in state X_T^Ψ.

This problem is similar to before, except now we want to find a policy Ψ that maximises EMV. Specifically, for some initial state i_0, we want an optimal policy

$$\Psi^* = \{f_0^*, f_1^*, \ldots, f_{T-1}^*\}$$

that satisfies

$$V^*(i_0) = \max_{\Psi \in \Gamma} V_0^\Psi(i_0),$$

where Γ is the set of all admissible policies.

Stage $T = 4$:

If we assume there is no advantage to be gained by finishing in any particular state, then $V_4^*(1) = V_4^*(2) = 0$.

Stage $T - 1 = 3$:

$$\begin{aligned}
v(1, \text{NO}) &= (0.5 \times 9 + 0.5 \times 3) + (0.5 \times V_4^*(1) + 0.5 \times V_4^*(2)) \\
&= (0.5 \times 9 + 0.5 \times 3) + 0 = 6, \\
v(1, \text{DES}) &= 0.75 \times 5 + 0.25 \times (-1) = 3.5, \\
v(2, \text{NO}) &= 0.3 \times 3 + 0.7 \times (-4) = -1.9, \\
v(2, \text{DES}) &= 0.7 \times (-1) + 0.3 \times (-8) = -3.1.
\end{aligned}$$

Hence,

$$\begin{aligned}
V_3^*(1) &= \max\{v(1, \text{DES}), v(1, \text{NO})\} = 6, \quad \text{under NO DESIGN}, \\
V_3^*(2) &= \max\{v(2, \text{DES}), v(2, \text{NO})\} - 1.9, \quad \text{under NO DESIGN}.
\end{aligned}$$

Stage $T - 2 = 2$:

$$\begin{aligned}
v(1, \text{NO}) &= (0.5 \times 9 + 0.5 \times 3) + (0.5 \times 6 + 0.5 \times (-1.9)) = 8.05, \\
v(1, \text{DES}) &= (0.75 \times 5 + 0.25 \times (-1)) + (0.75 \times 6 + 0.25 \times (-1.9)) = 7.53, \\
v(2, \text{NO}) &= (0.3 \times 3 + 0.7 \times (-4)) + (0.3 \times 6 + 0.7 \times (-1.9)) = -1.43, \\
v(2, \text{DES}) &= (0.7 \times (-1) + 0.3 \times (-8)) + (0.7 \times 6 + 0.3 \times (-1.9)) = 0.53.
\end{aligned}$$

Hence,

$$
\begin{aligned}
V_2^*(1) &= 8.05, \quad \text{under NO DESIGN} \\
V_2^*(2) &= 0.53, \quad \text{under DESIGN,} \\
V_1^*(1) &= 10.29, \quad \text{under NO DESIGN} \\
V_1^*(2) &= 2.69, \quad \text{under DESIGN.} \\
V_0^*(1) &= 12.49, \quad \text{under NO DESIGN} \\
V_0^*(2) &= 4.91, \quad \text{under DESIGN.}
\end{aligned}
$$

In summary, we have:

i	$V_0^*(i)$	$f_0^*(i)$	$V_1^*(i)$	$f_1^*(i)$	$V_2^*(i)$	$f_2^*(i)$	$V_3^*(i)$	$f_3^*(i)$	$V_4^*(i)$
1	12.49	NO	10.29	NO	8.05	NO	6.00	NO	0
2	4.91	DES	2.69	DES	0.53	DES	−1.90	NO	0

Table 3.2: The Optimal Policy

This indicates that

- Never engage in design work if the company is thriving.
- If the company is floundering, do design work if in stages 0, 1 or 2. (Equivalently, if the company is floundering, do design work if there is more than 1 stage-to-go.)

§3.4 INDEFINITE HORIZON PROBLEMS

STOCHASTIC SHORTEST PATH MDP (SPMDP)

We now consider one type of problems where the horizon T is not deterministically known in advance, but is a random variable. In particular, we investigate a type of MDPs where the underlying Markov chain has an **absorbing state** and where every transition incurs some cost. This implies there are associated costs that are added at each decision stage. **Our aim** will be to find a policy that minimises the expected total costs, from all possible starting states, until absorption occurs.

In the two previous examples (one minimisation and one maximisation), we analysed policies of the form $\Psi = \{f_0, f_1, \ldots, f_{T-1}\}$, which allowed us the flexibility of specifying a different action at each stage. These are **deterministic Markov policies**, where the qualifier **determistic** refers to the fact that we choose precisely **one** action to take at each state i and each time t. A **randomised Markov policy** specifies one or more actions u_k to take at each state i and each time t, with associated probabilities that add up to 1.

In this study of SPMDPs, we restrict ourselves to the consideration of policies of the form

$$\Psi = \{f, f, \ldots, f\},$$

which are **stationary** and usually denoted by $\Psi = f \in \Gamma$.

Defn 3.4. *A **Stochastic Shortest Path MDP** with a finite state space and finite set of actions is defined as follows. There are a finite number of states $i = 1, \dots N$, and an extra absorbing state d. For each state i, there is a finite set of actions $\mathscr{U}(i)$ that can be selected. At the absorbing state d, no action needs to be chosen. Transitions from state i to state j are governed by the probabilities $p_{ij}(u)$, where $u \in \mathscr{U}(i)$. So the state transition probabilities at state i depend on the particular action $u \in \mathscr{U}(i)$ that is selected. Transitions from state i to state j incur a cost r_{ij}.*

The expected single-step cost incurred by selecting action u when in state i is therefore given by

$$\mathbb{E}[c(i,u)] = \sum_{j=1}^{N} p_{ij}(u) r_{ij}. \tag{3.4}$$

***Example* 37** (A simple stochastic shortest path problem)**.** A guide, Maizura, is standing on top of a mountain, looking down at a hiker, Ross, who is trying to get HOME, as quickly as possible. Consider Figure 3.3, depicting three locations (states), 1, 2 and 3.

States 1 and 2 are on opposite sides of a lake, and are connected by a bridge, which Ross takes 1 minute to traverse. State 3 is HOME, the destination/termination state/absorbing state. There is a trail directly joining state 1 to the destination state 3, but it passes through a swamp, and takes 2 minutes to traverse. There is a paved road directly joining state 2 to state 3, which only takes 0.7 minutes to traverse.

The actions that Maizura can take when Ross is at 1 or 2, are to yell CROSS THE BRIDGE, or to yell FOLLOW THE LAKE.

Suppose also that Ross has probability 1/4 of not hearing Maizura's instruction correctly, and in fact doing the opposite of what he is told. Assume Ross never remains in the same location after receiving an instruction, which

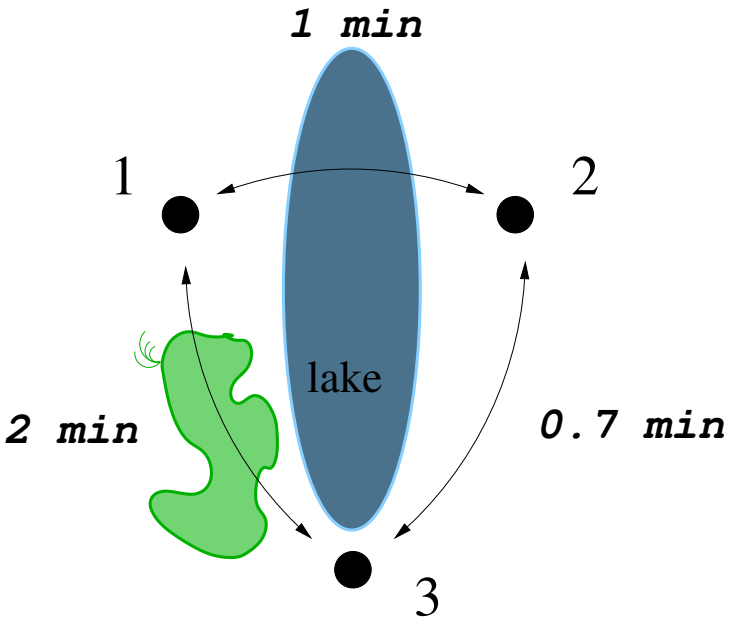


Figure 3.3: *Hiker Problem.*

means he always moves to another location. We are given the transition costs (time)

$$r_{12} = r_{21} = 1, \quad r_{13} = 2, \quad r_{23} = 0.7.$$

Which instruction should Maizura yell out to Ross at each location? There are four possible stationary policies for this problem:

$$f_1 = (u_1(1), u_1(2)) = (\text{FOLLOW LAKE}, \text{ FOLLOW LAKE}),$$
$$f_2 = (u_2(1), u_2(2)) = (\text{FOLLOW LAKE}, \text{ CROSS BRIDGE}),$$
$$f_3 = (u_3(1), u_3(2)) = (\text{CROSS BRIDGE}, \text{ FOLLOW LAKE}),$$
$$f_4 = (u_4(1), u_4(2)) = (\text{CROSS BRIDGE}, \text{ CROSS BRIDGE}).$$

The general form for the transition probability matrix corresponding to

the policy f_k is

$$P_{f_k} = \begin{pmatrix} p_{11}(f_k(1)) & p_{12}(f_k(1)) & p_{13}(f_k(1)) \\ p_{21}(f_k(2)) & p_{22}(f_k(2)) & p_{23}(f_k(2)) \\ 0 & 0 & 1 \end{pmatrix}.$$

Corresponding to each of the four possible policies are

$$P_{f_1} = \begin{pmatrix} 0 & \frac{1}{4} & \frac{3}{4} \\ \frac{1}{4} & 0 & \frac{3}{4} \\ 0 & 0 & 1 \end{pmatrix}, \qquad P_{f_2} = \begin{pmatrix} 0 & \frac{1}{4} & \frac{3}{4} \\ \frac{3}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 1 \end{pmatrix},$$

$$P_{f_3} = \begin{pmatrix} 0 & \frac{3}{4} & \frac{1}{4} \\ \frac{1}{4} & 0 & \frac{3}{4} \\ 0 & 0 & 1 \end{pmatrix}, \qquad P_{f_4} = \begin{pmatrix} 0 & \frac{3}{4} & \frac{1}{4} \\ \frac{3}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 1 \end{pmatrix}.$$

We will solve the problem in Example 37 by evaluating the costs associated with all possible policies, and identifying the policy that simultaneously minimises the expected cost from both of the possible starting states 1 and 2. Note that in this case, the cost measure is expressed in units of time, so that the total cost corresponds to the total amount of time taken for the hiker to reach the destination state. Of course, the cost could correspond to some other quantity, such as the amount of energy expended.

Let $J^{f_k}(i)$ be the expected cost incurred by Ross in reaching the destination state 3 from state i, given that Maizura follows the policy f_k. Employing a first step analysis, we obtain the optimality equations

$$J^{f_k}(i) = \sum_{j=1}^{3} p_{ij}(f_k(i)) \left(r_{ij} + J^{f_k}(j) \right) \quad \text{for } i = 1,2, \tag{3.5}$$

with the boundary condition $J^{f_k}(3) = 0$ for all policies $f_k, k = 1,\ldots,4$. This boundary condition makes sense physically, because the destination state is an absorbing state at which the Ross incurs no further cost.

The above system of equations can also be written

$$J^{f_k}(i) = c\left(i, f_k(i)\right) + \sum_{j=1}^{3} p_{ij}(f_k(i)) J^{f_k}(j) \quad \text{for } i = 1,2,$$

where the expected single-stage transition cost at state i under the stationary policy f_k is

$$c(i, f_k(i)) = \sum_{j=1}^{3} p_{ij}(f_k(i)) r_{ij}.$$

We can then substitute the appropriate transition probabilities and state transition costs into (3.5) for each of the four policies, to obtain a system of equations.

For example, if $k = 1$,

$$J^{f_1}(1) = \frac{1}{4}(1) + \frac{3}{4}(2) + \frac{1}{4}J^{f_1}(2) + \frac{3}{4}J^{f_1}(3) = \frac{7}{4} + \frac{1}{4}J^{f_1}(2),$$
$$J^{f_1}(2) = \frac{1}{4}(1) + \frac{3}{4}(0.7) + \frac{1}{4}J^{f_1}(1) + \frac{3}{4}J^{f_1}(3) = \frac{3.1}{4} + \frac{1}{4}J^{f_1}(1).$$

Solving gives us

$$J^{f_1}(1) = 2.07\overline{333}, \quad J^{f_1}(2) = 1.29\overline{333}.$$

Similarly solving for all policies and states, we get the solutions shown in Table 3.3. The optimal policy, which minimises cost, is for the guide to always yell the instruction FOLLOW LAKE, which corresponds to f_1.

Note that if the Ross could hear the guide perfectly, thus always following the exact instruction, the optimal policy would be f_3. This can be deduced by inspection of the diagram, since the optimal path from state 2 to the destination is directly along the lake, and the optimal path from state 1 to the destination is to first cross the bridge to state 2 (and then to follow the lake to the destination).

k	$f_k(1)$	$f_k(2)$	$J^{f_k}(1)$	$J^{f_k}(2)$
1	FOLLOW LAKE	FOLLOW LAKE	2.07	1.29
2	FOLLOW LAKE	CROSS BRIDGE	2.44	2.75
3	CROSS BRIDGE	FOLLOW LAKE	2.25	1.34
4	CROSS BRIDGE	CROSS BRIDGE	4.44	4.26

Table 3.3: The stationary policies and corresponding costs

We have just solved the above stochastic shortest path MDP by enumerating all possible policies, evaluating their associated expected costs and identifying the policy that yields the minimum expected cost. This **exhaustive search** approach is clearly not feasible if the number of possible policies is large and so other techniques are needed for such problems (see later).

§3.5 Infinite Horizon Problems

There are two types of Infinite Horizon MDPs: Average Value (Cost) MDPs, and Discounted MDPs.

Infinite Horizon Average Value (Cost) MDPs

We now reconsider Example 36, where the company intends to operate indefinitely, and hence we do not have a predetermined value for T.

Observe that in the **Value Iteration**, the increase in $V_t^*(i)$ between subsequent years appeared to be converging to a constant value, which we will assume exists and call it the **gain** g. This gain represents the average value of remaining in business for a year after some **relatively long time** and must be the same for all initial states $i \in \mathcal{S}$.

In Example 36:

t	1	2	3	4
$V^*_{t-1}(1) - V^*_t(1)$	2.20	2.24	2.05	6.00
average $\updownarrow$	2.21	2.20	2.24	2.05
$V^*_{t-1}(2) - V^*_t(2)$	2.22	2.16	2.43	-1.90

In general, for a problem with n states, we can find the exact value of g by considering the transition matrix corresponding to the optimal policy.

First, note that finite-state Markov chains always have a stationary distribution. Let the stationary distribution of the transition matrix corresponding to the optimal policy be given by

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots \pi_n).$$

Then, π_j is the probability that after a relatively long time and under the optimal policy the system will be in state j. The gain is therefore the sum, over all possible states, of the products of the probability of being in each state and the expected reward over a single stage commencing in that state.

As the expected reward for starting in state $i \in \{1, 2, \ldots, n\}$ is given by

$$q_i = \sum_{j=1}^{n} p_{ij} r_{ij},$$

the gain is therefore

$$g = \sum_{i=1}^{n} \pi_i q_i.$$

When the company intends to operate indefinitely, it can be established that the optimal steady state policy is to DESIGN if the state of the company is floundering and NOT TO DESIGN otherwise. The steady state policy has the following transition and reward matrices

$$P_{opt} = \begin{pmatrix} 0.5 & 0.5 \\ 0.7 & 0.3 \end{pmatrix}, \quad R_{opt} = \begin{pmatrix} 9 & 3 \\ -1 & -8 \end{pmatrix}.$$

Solving the equilibrium equations $\boldsymbol{\pi} P_{opt} = \boldsymbol{\pi}$, we obtain

$$\boldsymbol{\pi} = \left(\frac{7}{12}, \frac{5}{12}\right),$$

which leads to

$$g = \frac{7}{12}(0.5 \times 9 + 0.5 \times 3) + \frac{5}{12}(0.7 \times (-1) + 0.3 \times (-8)) = 2.208.$$

POLICY ITERATION

The existence of a constant in steady state is the essence of something known as **Policy Iteration**, where we have problems that do not have a pre-determined value for T, but have a time-to-go that is infinite.

We have been performing backward recursions and calculating values at each stage, a process known as **Value iteration**. In the infinite horizon case, these values may become positively or negatively infinite. However, we observed that even though

$$\lim_{T\to\infty} \mathbb{E}\left[\sum_{n=0}^{T-1} v\left(X_n, f_n\left(X_n\right) | X_0 = i\right)\right] \to \infty,$$

it may be true that

$$\lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{n=0}^{T-1} v\left(X_n, f_n\left(X_n\right) | X_0 = i\right)\right] \to g, \tag{3.6}$$

where $-\infty < g < \infty$.

Policy Iteration Procedure

Consider that for a chosen policy, we have some finite horizon T with

$$V_k(i) = \sum_{j=1}^{n} p_{ij}\left(r_{ij} + V_{k+1}(j)\right) \tag{3.7}$$

and so for $T \gg k$, we have

$$V_k(i) = g(T-k) + \varphi(i), \tag{3.8}$$

where $\varphi(i)$ is the value of starting in state i at time k. One of the consequences of (3.8) is that

$$V_k(i) = g + V_{k+1}(i), \tag{3.9}$$

which implies that the specific value of starting in state i given by $\varphi(i)$ becomes negligible over time. Substituting (3.8) into (3.7) yields

$$\begin{aligned} (T-k)g + \varphi(i) &= \sum_{j=1}^{n} p_{ij}\left(r_{ij} + (T-k-1)g + \varphi(j)\right) \\ \Leftrightarrow \quad (T-k)g + \varphi(i) - (T-k-1)g &= \sum_{j=1}^{n} p_{ij}\left(r_{ij} + \varphi(j)\right) \\ \Leftrightarrow \varphi(i) + g &= \left(\sum_{j=1}^{n} p_{ij} r_{ij}\right) + \left(\sum_{j=1}^{n} p_{ij}\varphi(j)\right). \end{aligned} \tag{3.10}$$

Equations (3.10) represent **the value determination operation**. The **Policy Improvement Routine** for each state i, is to find an alternative policy that maximises the expression on the RHS of (3.10).

Policy Improvement Routine (PIR)

Given the $\varphi(j)$ for the current policy, we want to find an alternative policy f' that maximises

$$\left(\sum_{j=1}^{n} p'_{ij} r'_{ij}\right) + \left(\sum_{j=1}^{n} p'_{ij}\varphi(j)\right), \tag{3.11}$$

where p'_{ij} and r'_{ij} are, respectively, the probability transitions and rewards under the alternative policy φ'.

The policy iteration procedure revolves around (3.10) and (3.11). It is convenient to start with (3.10) using any policy (for example, a policy maximize immediate reward). Note that Equations (3.10) are a set of n linear equations in $n+1$ unknowns $\varphi(1), \varphi(2), \ldots, \varphi(n)$ and g. A unique solution may be found by setting $\varphi(1) = 0$, so that the other unknowns can be found and which represent relative values of starting in states $i \neq 1$. The next step is to apply the PIR criteria using (3.11).

The PIR will generally not immediately give the optimal policy unless our first guess (initial policy) was an exceptionally good choice, so that there is no alternative policy such that (3.11) yields an improved value. We will show later that the PIR can only yield an alternative policy that is an improvement. The procedure itself, which iterates between possible policies, stops or terminates when two policies are identical in subsequent iterations. Note that the **PIR** may also be used for minimisation problems.

Policy Improvement Routine

- Step 1. Start with any policy.
- Step 2. Solve

$$\phi(i) + g = \left(\sum_{j=1}^{n} p_{ij} r_{ij}\right) + \left(\sum_{j=1}^{n} p_{ij} \phi(j)\right) \quad \text{to find } \varphi(i) \text{ and } g.$$

- Step 3. Find an alternative policy f' and see if it improves

$$\left(\sum_{j=1}^{n} p'_{ij} r'_{ij}\right) + \left(\sum_{j=1}^{n} p'_{ij} \phi(j)\right)$$

- Step 4. If it does not, STOP. Otherwise, set $p_{ij} = p'_{ij}$, $r_{ij} = r'_{ij}$ and go back to Step 2.

***Example* 38** (Designer Furniture Manufacturer)**.** We revisit the company and consider NO DESIGN as our initial policy.

Iteration 1: Using Equation (3.10) gives us the system

$$g+\varphi(1) = 0.5\times 9 + 0.5\times 3 + 0.5\varphi(1) + 0.5\varphi(2),$$
$$g+\varphi(2) = 0.3\times 3 + 0.7\times(-4) + 0.3\varphi(1) + 0.7\varphi(2).$$

Setting $\varphi(1) = 0$, we obtain, for $i = 1$ and for $i = 2$, respectively,

$$g = 0.5\times 9 + 0.5\times 3 + 0.5\varphi(2), \tag{3.12}$$
$$g+\varphi(2) = 0.3\times 3 + 0.7\times(-4) + 0.7\varphi(2). \tag{3.13}$$

The difference (3.12) – (3.13) removes the gain g and gives

$$\varphi(2) = 0.9 - 2.8 - 4.5 - 1.5 + 0.2\varphi(2) \Rightarrow \varphi(2) = -9.875.$$

Under DESIGN, using $\varphi(1) = 0$, $\varphi(2) = -9.875$ and (3.10) leads to

$$3.75 - 0.25 + 0 + (0.25)(-9.875) = 1.0312 \quad \text{for } i = 1,$$
$$-0.7 - 2.4 + 0 + (0.3)(-9.875) = -6.0625 \quad \text{for } i = 2.$$

As $1.0625 > 1.0312$ and $-6.0625 > -8.8125$, our new policy is NO DESIGN if in state $i = 1$, and DESIGN if in state $i = 2$.

Iteration 2: Under this new policy, setting $\varphi(1) = 0$, Equation (3.10) gives

$$g = (0.5)(9) + (0.5)(3) + 0.5\varphi(2), \tag{3.14}$$
$$g+\varphi(2) = (0.7)(-1) + (0.3)(-8) + 0.3\varphi(2). \tag{3.15}$$

The difference (3.14) – (3.15) yields

$$\varphi(2) = -0.7 - 2.4 - 4.5 - 1.5 - 0.2\varphi(2) \quad \Rightarrow \varphi(2) = -7.5833.$$

With $\varphi(1) = 0$, $\varphi(2) = -7.5833$, under NO DESIGN

$$4.5 + 1.5 + 0 + (0.5)(-7.5833) = 2.2083 \quad \text{for } i = 1,$$
$$0.9 - 2.8 + 0 + (0.7)(-7.5833) = -7.2083 \quad \text{for } i = 2.$$

and under DESIGN

$$3.75 - 0.25 + 0 + (0.25)(-7.5833) = 1.6042 \quad \text{for } i = 1,$$
$$-0.7 - 2.4 + 0 + (0.3)(-7.5833) = -5.3750 \quad \text{for } i = 2.$$

As $2.2083 > 1.6042$ and $-5.3750 > -7.2083$, our new policy is NO DESIGN if in state $i = 1$, and DESIGN if in state $i = 2$. This is identical to the previous policy, and so we stop the iterations as the optimal policy has been found.

This optimal policy is what we might intuitively conclude from the previous investigation for the problem with $T = 4$, where the company only did design work if it was FLOUNDERING and there was more than 1 stage-to-go. (There is always more than 1 stage-to-go in the infinite horizon case!)

Note that the policy just discovered is a stationary policy, and gives us the action to be taken when in each of the states of the process *independent* of the time that we are in each of these states.

Defn 3.5 (Average value/cost MDP)**.** *Consider an MDP with a finite number of states, for which the underlying Markov chain is positive recurrent, and the termination time T is effectively infinite. In this case, each state is visited an infinite number of times.*

If the expected value (cost) of all single-stage transitions is strictly positive, it is clear that the total value (cost) of any policy Ψ, is infinite, which tells us nothing about the different policies.

*However, the **average value (cost) per stage** g is in many cases well defined and given by*

$$g = \lim_{T\to\infty} \frac{1}{T} \mathbb{E}\left[\sum_{n=0}^{T} v(X_n, f_n(X_n))\right].$$

*An MDP that finds an optimal policy that maximises (minimises) g is known as an **average value (cost) MDP.***

DISCOUNTED MDPS

Let $\beta \in (0,1]$ be a *discount factor*, which acts as a measure of the importance that we place on costs incurred into the future, as opposed to costs incurred at the present time. As a practical example, consider an asset whose value depreciates over time. In a decision process involving such an asset, we must factor in this depreciation when we evaluate the costs of different policies. We can define the cost criterion factoring in $\beta \in (0,1]$ over a finite horizon T as

$$\sum_{n=0}^{T-1} \beta^n \, c(X_n, u_n) + \beta^T C(X_T).$$

Suppose the costs are uniformly bounded, such that

$$|c(X_n, u_n)| < B \quad \text{for all } X_n \in \mathscr{S} \text{ and } n \geq 0.$$

Then, if discounting is strict so that $0 < \beta < 1$, the infinite horizon cost is bounded by $B/(1-\beta)$, which we can show by using geometric series.

The minimal present value $J_k^\Psi(i)$ at time k of expected cost from time k up to time T, starting in state i at time k under policy Ψ, is given by

$$\inf_\Psi \mathbb{E}\left[\sum_{n=k}^{T-1} \beta^{n-k} c\left(X_n^\Psi, f_n\left(X_n^\Psi\right)\right) + \beta^{T-k} C(X_T) \mid X_k = i\right]. \tag{3.16}$$

We then obtain the optimality equations

$$J_T(i) = C(X_T), \tag{3.17}$$

$$J_k(i) = \inf_{u_k \in \mathscr{U}_k(i)} \left\{\mathbb{E}[c(i,u_k)] + \beta \mathbb{E}\left[J_{k+1}(X_{k+1})\right]\right\} \quad \text{for } k \leq T. \tag{3.18}$$

The s-horizon cost $F_s^\Psi(i)$ under policy Ψ

In the finite horizon case, the value function can simply be obtained from (3.17) and (3.18) using backward recursions from the terminal time T.

Consider a stationary policy Ψ, where $c(i,u_k) = c(i,u)$ for all $k \geq 0$, and assume that $C(X_T) = 0$ for all final states, so that we can define the **s-horizon cost $F_s^\Psi(i)$ under policy Ψ:**

$$F_s^\Psi(i) = \mathbb{E}\left[\sum_{n=0}^{s-1} \beta^n c(X_n,u) \middle| X_0 = i\right]. \tag{3.19}$$

The subscript s means that we are looking at the cost in terms of s steps-to-go. If we take the infimum with respect to Ψ, we obtain the optimal s-horizon cost

$$F_s(i) = \inf_{\Psi\in\Gamma} F_s^\Psi(i).$$

Note that this infimum always exists and satisfies

$$F_s(i) = \inf_u \left\{\mathbb{E}[c(i,u)] + \beta\mathbb{E}\left[F_{s-1}(X_1) \mid X_0 = i\right]\right\},$$

with terminal condition $F_0(i) = 0$ for all i.

The infinite-horizon cost $F^\Psi(i)$ under policy Ψ

We now consider the case where the discounted MDP has an infinite horizon, or where $s \to \infty$. The **infinite horizon cost $F^\Psi(i)$ under policy Ψ** is given by

$$F^\Psi(i) = \lim_{s\to\infty} F_s^\Psi(i). \tag{3.20}$$

The limit in Equation (3.20) need not always exist, so we will now investigate conditions under which it does. To this end, we will formulate everything in terms of the minimisation problem.

Defn 3.6. *Minimisation problems can be classified into three categories.*

D: *Discounted Programming.*
$0 < \beta < 1$
and $|c(i,u)| < B$ *for all* $i \in \mathscr{S}$ *and* $u \in \mathscr{U}(i)$.

N: *Negative Programming.*
$0 < \beta \leq 1$
and $c(i,u) \geq 0$ *for all* $i \in \mathscr{S}$ *and* $u \in \mathscr{U}(i)$.

P: *Positive Programming.*
$0 < \beta \leq 1$
and $c(i,u) \leq 0$ *for all* $i \in \mathscr{S}$ *and* $u \in \mathscr{U}(i)$.

The terms **positive** and **negative** arises from considering that

- maximising positive rewards (P) is equivalent to minimising negative costs;
- maximising negative rewards (N) is equivalent to minimising positive costs.

In cases of N and P, wlog, we will usually take $\beta = 1$.

The existence of the limit in (3.20) (possibly $\pm\infty$) is assured in cases N and P by monotone convergence and in the case D because the total cost occurring after the sth step is bounded by

$$\beta^s \frac{B}{1-\beta} \qquad \text{and} \qquad \lim_{s\to\infty} \beta^s \frac{B}{1-\beta} \to 0.$$

The infimal infinite-horizon cost is defined by

$$F(i) = \inf_{\Psi\in\Gamma} F^{\Psi}(i) = \inf_{\Psi\in\Gamma} \lim_{s\to\infty} F_s^{\Psi}(i). \tag{3.21}$$

Theorem 3.3. *Suppose D, N or P hold, then*

$$F(i) = \inf_u \left\{ \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F(X_1) \mid X_0 = i, u_0 = u\right] \right\}. \tag{3.22}$$

Proof. First, we show that $F(i) \geq$ RHS of (3.22), and then $F(i) \leq$ RHS of (3.22).

(a) **To show** $F(i) \geq$ RHS of (3.22):
Suppose Ψ is a policy that chooses $u_0 = u$ when $X_0 = i$, then

$$F_s^{\Psi}(i) = \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F_{s-1}^{\Psi}(X_1) \mid X_0 = i, u_0 = u\right] \tag{3.23}$$

Now, either D, N or P is sufficient to allow us to take limits on both sides of (3.23) and interchange the order of limit and expectation. In cases N and P, this is because of monotone convergence, where infinity is allowed as a possible limiting value.

We obtain the following

$$\begin{aligned} F^{\Psi}(i) &= \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F^{\Psi}(X_1) \mid X_0 = i, u_0 = u\right] \\ &\geq \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F(X_1) \mid X_0 = i, u_0 = u\right] \\ &\geq \inf_u \left\{ \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F(X_1) \mid X_0 = i, u_0 = u\right] \right\}. \end{aligned}$$

Taking the infinimum the LHS over $\Psi \in \Gamma$ gives us $F(i) \geq$ RHS of (3.22).

(b) **To show** $F(i) \leq$ RHS of (3.22): We consider another policy Ψ^* that after choosing some u_0 when $X_0 = i$, reaches state X_1 and chooses a different policy Ψ_1, which is suboptimal by less than some ε from then on; that is,

$$F^{\Psi_1}(X_1) \leq F(X_1) + \varepsilon.$$

Such a policy must exist by definition of F, although Ψ_1 will depend on X_1. Thus, we have

$$\begin{aligned} F(i) &\leq F^{\Psi^*}(i) \\ &= \mathbb{E}[c(i,u)] + \beta\mathbb{E}\left[F^{\Psi_1}(X_1) \mid X_0 = i, u_0 = u\right] \\ &\leq \mathbb{E}[c(i,u)] + \beta\mathbb{E}\left[F(X_1) + \varepsilon \middle| X_0 = i, u_0 = u\right] \\ &\leq \mathbb{E}[c(i,u)] + \beta\mathbb{E}\left[F(X_1) \mid X_0 = i, u_0 = u\right] + \beta\varepsilon \end{aligned}$$

which implies

$$F(i) \leq \inf_u \left\{\mathbb{E}[c(i,u)] + \beta\mathbb{E}\left[F(X_1) \mid X_0 = i, u_0 = u\right]\right\} + \beta\varepsilon.$$

Recalling that ε is arbitrary yields the result. □

***Example* 39** (Factory)**.** At a factory, a machine can be operated in either AUTOMATIC MODE or MANUAL MODE. If it is operated in AUTOMATIC MODE on a particular day, there is a probability q that it will fail to start the next day. Similarly, if if is operated in MANUAL MODE, there is a probability $p < q$ (otherwise why would you operate manually?) that it will fail to start the next day.

Running the machine in AUTOMATIC MODE costs \$0 per day, while running in MANUAL MODE costs \$$c$ per day for the attendant. If the machine fails to start on a particular day, it must be repaired, a process that takes the entire day at a cost of \$8c, due to lost production, labour, parts etc. Costs are discounted by a factor $\beta \in (0,1)$.

We would like to minimise the expected discounted cost over an essentially infinite horizon. Let $F(0)$ and $F(1)$ denote the minimal costs from NOT WORKING in the morning and WORKING in the morning respectively. Then, the state space is $\mathscr{S} = \{\text{NOT WORKING}, \text{WORKING}\} \equiv \{0,1\}$.

If we want to show that it is optimal to run the machine continuously in automatic mode rather than continuously in manual mode, we need to determine some conditions based on the parameterisations that we have. That

is, using the two possible modes (actions) {MANUAL, AUTOMATIC}, Theorem 3.3 says that we can write down the following

$$F(0) = \inf\{8c + \beta\left[pF(0) + (1-p)F(1)\right],\ 8c + \beta\left[qF(0) + (1-q)F(1)\right]\},$$
$$F(1) = \inf\{c + \beta\left[pF(0) + (1-p)F(1)\right],\ \beta\left[qF(0) + (1-q)F(1)\right]\}.$$

Then operating in MANUAL MODE only we have

$$F^M(0) = 8c + \beta\left[pF^M(0) + (1-p)F^M(1)\right], \tag{3.24}$$
$$F^M(1) = c + \beta\left[pF^M(0) + (1-p)F^M(1)\right]. \tag{3.25}$$

The difference (3.24)-(3.25) gives us

$$F^M(0) - F^M(1) = 7c. \tag{3.26}$$

Subsequently using (3.26) in (3.24) and (3.25) leads to

$$F^M(0) = \frac{c\big(8 - 7\beta(1-p)\big)}{1-\beta}, \quad F^M(1) = \frac{c(1+7\beta p)}{1-\beta}.$$

Similarly, operating in AUTOMATIC MODE only it can be shown that

$$F^A(0) = \frac{8c\big(1 - \beta(1-q)\big)}{1-\beta}, \quad F^A(1) = \frac{8c\beta q}{1-\beta}.$$

If running continuously in AUTOMATIC MODE is better, we must have

$$\begin{aligned} & F^A(1) \le F^M(1) \\ \Leftrightarrow \quad & \frac{8c\beta q}{(1-\beta)} < \frac{c(1+7\beta p)}{(1-\beta)} \\ & \Leftrightarrow \beta \le \frac{1}{(8q-7p)}, \end{aligned}$$

since $8q - 7p > 0$. Again, if running continuously in AUTOMATIC MODE is better, we must also have that

$$\begin{aligned} & F^A(0) \leq F^M(0) \\ \Leftrightarrow \quad & \frac{8c\big(1-\beta(1-q)\big)}{\big(1-\beta\big)} \leq \frac{c\big(8-7\beta(1-p)\big)}{\big(1-\beta\big)} \\ \Leftrightarrow \quad & -8(1-q) \leq -7(1-p) \\ \Leftrightarrow \quad & (8q-7p) \leq 1, \end{aligned}$$

which means $0 \leq 8q - 7p \leq 1$.

We should of course check other stationary policies to find the optimal policy. That is, cross over policies, where we run in one mode if in one state and the other mode if in the other state (see later).

§3.6 POSITIVE PROGRAMMING & THE VALUE ITERATION ALGORITHM

POSITIVE PROGRAMMING

Positive programming concerns maximising non-negative rewards or minimising non-positive costs and in some cases, there is no optimal policy.

***Example* 40** (No optimal policy.)**.** Suppose that $\mathscr{S} = \mathbb{Z}^+$, the set of non-negative integers. When in state i, we have a choice of either

- moving to state $i+1$ and receive no reward, or
- moving to state 0, receiving a reward $(1 - 1/i)$, and then remaining in state 0 thereafter with no further reward.

The optimality equation may be written

$$F(i) = \sup\left\{\left(1-\frac{1}{i}\right), F(i+1)\right\} \quad \text{for } i > 0.$$

Clearly, $F(i) = 1, i > 0$, but the policy that chooses the maximising action always moves up one state and hence receives zero reward. Hence, there is no policy that yields the reward of 1.

CHARACTERISATION OF THE OPTIMAL POLICY

The following theorem provides a necessary and sufficient condition for a policy to be optimal.

Theorem 3.4. *Suppose that D or P holds, and that Ψ is a policy whose value function $F^{\Psi}(i)$ satisfies the optimality equation:*

$$F^{\Psi}(i) = \sup_u \left\{ \mathbb{E}[r(i,u)] + \beta \mathbb{E}\left[F^{\Psi}(X_1) \mid X_0 = i, u_0 = u \right] \right\}.$$

Then Ψ is the optimal policy. (Note: This is a maximisation problem.)

Proof. Let Ψ_1 be any other policy and suppose it takes $u_t(i) = f_t(i)$. Since $F^{\Psi}(i)$ satisfies the optimality equation,

$$\begin{aligned} F^{\Psi}(i) &= \sup_u \left\{ \mathbb{E}[r(i,u)] + \beta \mathbb{E}\left[F^{\Psi}(X_1) \mid X_0 = i, u_0 = u \right] \right\} \\ &\geq \underbrace{\mathbb{E}[r\left(i, f_0(i)\right)]}_{\text{expected single-stage reward under policy } \Psi_1} + \beta \mathbb{E}\left[F^{\Psi}(X_1) | X_0 = i, u_0 = f_0(i) \right]. \end{aligned}$$

By repeated substitution of this into itself, we find

$$F^{\Psi}(i) \geq \mathbb{E}_{\Psi_1}\left[\sum_{t=0}^{s-1} \beta^t r(X_t, u_t) \middle| X_0 = i \right] + \beta^s \mathbb{E}\left[F^{\Psi}(X_s) | X_0 = i \right], \tag{3.27}$$

where the subscript Ψ_1 in the first expectation on the RHS indicates that the actions u_t at each time $t = 0, \ldots, s-1$ are determined by the policy Ψ_1.

In the case P, we know that $0 < \beta \leq 1$ and that $c(i,u) \leq 0$, the latter implying $r(i,u) \geq 0$ (as $r(i,u) := -c(i,u)$). Consequently,

$$\beta^s \mathbb{E}\left[F^{\Psi}(X_s) | X_0 = i \right] \geq 0.$$

Hence, we have

$$F^{\Psi}(i) \geq \mathbb{E}_{\Psi_1}\left[\sum_{t=0}^{s-1} \beta^t r(X_t, u_t) | X_0 = i\right].$$

Letting $s \to \infty$ and interchanging the limit and the expectation operators on the RHS (which we are allowed to do because of monotone convergence!), we have

$$F^{\Psi}(i) \geq F^{\Psi_1}(i).$$

In the case D, we have $0 < \beta < 1$ and $c(i,u)$ being uniformly bounded. Thus, we can directly let $s \to \infty$ and observe that

$$\lim_{s\to\infty} \beta^s \mathbb{E}\left[F^{\Psi}(X_s) | X_0 = i\right] = 0,$$

which implies that

$$F^{\Psi}(i) \geq F^{\Psi_1}(i).$$

□

***Example* 41** (Optimal gambling)**.** A gambler has $\$i$ and wants to increase this to $\$N$. At each stage they can bet any whole number of \$ not exceeding their capital, say $j \leq i$. If they win, with probability p, they now have $\$(i+j)$, or if they lose, with probability $q = (1-p) > 0$, they now have $\$(i-j)$. There are no draws, only wins or losses.

Suppose $p \geq 1/2$, and we have to prove that the **timid strategy** Ψ_1, of always betting only \$1, is optimal for the gambler to attain $\$N$. The optimality equation for maximising the probability of attaining $\$N$ is

$$F(i) = \max_{j, j\leq i} \left\{pF(i+j) + qF(i-j)\right\},$$

with $F(0) = 0$ and $F(N) = 1$.

To show that the timid strategy, Ψ_1, is optimal we need to find its value function, say $G(i) = F^{\Psi_1}(i)$, and then show that it is a solution to the optimality

equation. We have that $G(i) = pG(i+1)+qG(i-1)$, with $G(0) = 0$ and $G(N) = 1$ and trying a solution of the form $G(i) = w^i$ for some $w \in R$ yields the quadratic

$$\frac{w^i}{w^{i-1}} = p\,\frac{w^{i+1}}{w^{i-1}} + q\,\frac{w^{i-1}}{w^{i-1}} \Rightarrow (w-1)\left(w - \frac{q}{p}\right) = 0,$$

so that $w = 1$ or $w = \frac{q}{p}$, which for $p > \frac{1}{2}$ (by assumption) gives us

$$G(i) = A\left(\frac{q}{p}\right)^i + B, \quad \text{for } A, B \in R.$$

Using the boundary conditions

$$G(0) = 0 \Rightarrow B = -A,$$
$$G(N) = 1 \Rightarrow A = \frac{1}{\left(\frac{q}{p}\right)^N - 1},$$

we obtain

$$G(i) = \frac{1-(q/p)^i}{1-(q/p)^N}, \quad \text{for } p > \frac{1}{2}.$$

When $p = 1/2$, we use the boundary conditions in $G(i) = Ai + B$, to give us the final part of the problem, which yields

$$G(i) = \begin{cases} \dfrac{1-(q/p)^i}{1-(q/p)^N}, & p > \dfrac{1}{2}, \\ \dfrac{i}{N}, & p = \dfrac{1}{2}. \end{cases}$$

If $p = 1/2 = q$, then $G(i) = i/N$ clearly satisfies the optimality equation:

$$\max_{j,j\le i}\left\{\frac{1}{2}\left[\frac{i+j}{N}\right] + \frac{1}{2}\left[\frac{i-j}{N}\right]\right\} = \max_{j,j\le i}\left\{\frac{1}{2}\left[\frac{i+i}{N}\right] + \frac{1}{2}\left[\frac{j-j}{N}\right]\right\} = \frac{i}{N}.$$

If $p > 1/2$, we have to verify that $G(i) = \dfrac{1-(q/p)^i}{1-(q/p)^N}$ satisfies

$$G(i) = \max_{j, j \le i} \left\{ p \left[\frac{1-(q/p)^{i+j}}{1-(q/p)^N} \right] + q \left[\frac{1-(q/p)^{i-j}}{1-(q/p)^N} \right] \right\}. \tag{3.28}$$

Let W_j be the expression inside the braces $\{\ \}$ on the RHS of (3.28), then we can easily show that $W_{j+1} < W_j$, for $1 \le j \le i$. In particular, since $p > q$ and $p, q, N > 0$, this is equivalent to establishing that for all $1 \le k \le i$,

$$\begin{aligned} p\left(\frac{q}{p}\right)^{i+k} + q\left(\frac{q}{p}\right)^{i-k} &< p\left(\frac{q}{p}\right)^{i+k+1} + q\left(\frac{q}{p}\right)^{i-k-1} \\ \Rightarrow p\left(\frac{q}{p}\right)^{i+k}\left(1-\left(\frac{q}{p}\right)\right) &< q\left(\frac{q}{p}\right)^{i-k-1}\left(1-\left(\frac{q}{p}\right)\right) \\ \Rightarrow p\left(\frac{q}{p}\right)^{i+k} &< q\left(\frac{q}{p}\right)^{i-k-1} \end{aligned}$$

as all terms are positive because $1 > p > 1/2$.

Then,

$$\begin{aligned} p\left(\frac{q}{p}\right)^{i+k} &< q\left(\frac{q}{p}\right)^{i-k-1} \\ \Leftrightarrow \left(\frac{q}{p}\right)^{i-k-1}\left(p\left(\frac{q}{p}\right)^{2k+1} - q\right) &< 0 \\ \Leftrightarrow q\left(\left(\frac{q}{p}\right)^{2k} - 1\right) &< 0, \end{aligned}$$

which directly follows because $\left(\frac{q}{p}\right)^{2k} < 1$ for all $k > 0$.

Hence, $j = 1$ maximises the RHS of (3.28) as required and thus the timid strategy Ψ_1 maximises the probability of making $\$N$.

THE VALUE ITERATION ALGORITHM

Recall now the **Value Iteration method** that we used on finite horizon problems. Starting with $F_0(i) = 0$, we worked backwards iteratively for $s = 1, 2, \ldots$ to calculate

$$F_s(i) = \inf_{\Psi} \left\{ \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F_{s-1}^{\Psi}(X_1) \middle| X_0 = i, u_0 = u \right] \right\}.$$

So $F_s(i)$ is the infimal cost over s steps. Now let

$$F_\infty(i) = \lim_{s\to\infty} F_s(i) = \lim_{s\to\infty} \inf_{\Psi} F_s^{\Psi}(i) \tag{3.29}$$

The limit exists, either by monotone convergence under N or P, or by the fact that $\lim_{s\to\infty}$ of cost incurred $\to 0$ under D. Note that (3.29) reverses the order of $\lim\limits_{s\to\infty}$ and $\inf\limits_{\Psi}$ in (3.21), which is the infimal infinite-horizon cost

$$F(i) = \inf_{\Psi\in\Gamma} F^{\Psi}(i) = \inf_{\Psi\in\Gamma} \lim_{s\to\infty} F_s^{\Psi}(i).$$

The following theorem states that we can interchange the order of the operations and therefore $F_s(i) \to F(i)$. In the case N, we will need an additional assumption:

Defn 3.7 (Assumption A: Finite actions)**.** *Under assumption A, there are only finitely many possible values of u (finitely many actions) in each state.*

Theorem 3.5. *Suppose D or P holds, or N and Assumption A hold. Then, $F_\infty(i) = F(i)$.*

Proof. We prove first '≤', and then '≥' to get the result.

('≤') Given any policy $\overline{\Psi}$,

$$F_\infty(i) = \lim_{s\to\infty} F_s(i) = \lim_{s\to\infty} \inf_{\Psi} F_s^{\Psi}(i) \le \lim_{s\to\infty} F_s^{\overline{\Psi}}(i) = F^{\overline{\Psi}}(i).$$

Taking the infimum over $\overline{\Psi}$ gives $F_\infty(i) \le F(i)$.

('≥') **In the case** P, we have $c(i,u) \le 0$, so $F_s(i) \ge F(i)$. Now, let $s \to \infty$, then $F_\infty(i) \ge F(i)$.

In the case D, with $|c(i,u)| < B$, imagine subtracting $B > 0$ from every cost, which reduces the infinite-horizon cost under any policy by exactly $B/(1-\beta)$, and $F(i)$ and $F_\infty(i)$ also decrease by this amount. All costs are now negative, so the result above in the case P, again applies.

Alternatively, note that

$$F_s(i) - \beta^s B/(1-\beta) \le F(i) \le F_s(i) + \beta^s B/(1-\beta)$$

and hence $\lim_{s\to\infty} F_s(i) = F(i)$.

In the case N **with** A**,**

$$\begin{aligned} F_\infty(i) &= \lim_{s\to\infty} \min_u \left\{ c(i,u) + \mathbb{E}\left[F_{s-1}(X_1) | X_0 = i, u_0 = u\right] \right\} \\ &= \min_u \left\{ c(i,u) + \lim_{s\to\infty} \mathbb{E}\left[F_{s-1}(X_1) | X_0 = i, u_0 = u\right] \right\} \qquad (3.30) \\ &= \min_u \left\{ c(i,u) + \mathbb{E}\left[F_\infty(X_1) | X_0 = i, u_0 = u\right] \right\} \qquad (3.31) \end{aligned}$$

where Equality (3.30) follows because the minimum is over a finite number of terms (condition A), and Equality (3.31) follows by monotone convergence (since $F_s(i)$ increases in s).

Now let Ψ^* be the policy that chooses the minimising action on the RHS of (3.31). This implies, by substitution of (3.31) into itself, and using the fact that N implies $F_\infty \ge 0$,

$$\begin{aligned} F_\infty(i) &= \mathbb{E}_{\Psi^*}\left[\sum_{t=0}^{s-1} c(X_t, u_t) + F_\infty(X_s) | X_0 = i \right] \\ &\ge \mathbb{E}_{\Psi^*}\left[\sum_{t=0}^{s-1} c(X_t, u_t) | X_0 = i \right]. \end{aligned}$$

Letting $s \to \infty$ gives $F_\infty(i) \geq F^{\Psi^*}(i) \geq F(i)$, which leads to the required result $F_\infty(i) = F(i)$ if either D or P holds, or N and A hold. □

***Example* 42** (Factory revisited)**.** The s-horizon values are given by

$$\begin{aligned}
F_s(0) &= \inf\{F_s^M(0), F_s^A(0)\} \\
&= \inf\left\{ \begin{array}{l} 8c + \beta\left[pF_{s-1}^M(0) + (1-p)F_{s-1}^M(1)\right], \\ \\ \qquad 8c + \beta\left[qF_{s-1}^A(0) + (1-q)F_{s-1}^A(1)\right] \end{array} \right\} \\
F_s(1) &= \inf\{F_s^M(1), F_s^A(1)\} \\
&= \inf\left\{ \begin{array}{l} c + \beta\left[pF_{s-1}^M(0) + (1-p)F_{s-1}^M(1)\right], \\ \\ \qquad \beta\left[qF_{s-1}^A(0) + (1-q)F_{s-1}^A(1)\right] \end{array} \right\}
\end{aligned}$$

with $F_0(0) = \inf[0,0]$ and $F_0(1) = \inf[0,0]$.

We will use the MATLAB file **FactoryExample.m** on MyUni to investigate this algorithm. Effectively we have a function in the optimality equation which we can use to set up some recurrence relations that we can iterate on computer and investigate solutions for various policies. The function is monotonically increasing and because of the discounting parameter β, we would hope that the function will converge to the same constant values we calculated analytically. We can thus check our iterations using the analytic solutions for the fixed policies of running in either automatic mode or manual mode, as well as investigating other mixed policies.

§3.7 Negative Programming & Optimal Stopping

Recall that N, denoting Negative Programming, is about maximising non-positive rewards, $r(i,u) \leq 0$, or minimising non-negative costs, $c(i,u) \geq 0$, and that

1. a **Markov policy** is a policy that specifies the control at time t to be simply a function of the state and time. Earlier we used the notation

$$u_t(i) = f_t(i)$$

to specify the action at time t; convenient notation for a Markov policy.

2. If in addition the policy does not depend on time in its choice of action, then it is said to be **a stationary Markov policy**. We write $\Psi = (f, f, \ldots) = f$, where the stationary action $u = f(i)$.

CHARACTERISATION OF THE OPTIMAL POLICY

The following theorem gives a necessary and sufficient condition for a stationary policy to be optimal: namely, it must choose the optimal u on the RHS of the optimality equation. (Note that in this theorem we are requiring that the infimum over u is attained as the minimum over u, as would be the case if we make the finite action assumption, A.)

Theorem 3.6. *Suppose D or N holds. Suppose $\Psi = f$ is the stationary Markov policy such that*

$$f(i) = \operatorname*{argmin}_u \left\{ \mathbb{E}[c(i,u)] + \beta \mathbb{E}\left[F(X_1) | X_0 = i, u_0 = u\right] \right\}.$$

Then $F^{\Psi}(i) = F(i)$, and Ψ is optimal. That is, $u = f(i)$ is the stationary value of u that minimises the RHS of the optimality equation.

Proof. With $\Psi = f$, by substituting the optimality equation into itself and using the fact that Ψ specifies the minimising action at each stage,

$$F(i) = \mathbb{E}_{\Psi}\left[\sum_{t=0}^{s-1} \beta^t c(X_t, u_t) | X_0 = i \right] + \beta^s \mathbb{E}_{\Psi}\left[F(X_s) | X_0 = i\right]. \tag{3.32}$$

We now show that $F(i) \geq F^{\Psi}(i)$:

- In the case N, we can drop the final term on the RHS of (3.32) (because it is non-negative) and then let $s \to \infty$.

- In the case D, we can let $s \to \infty$ directly, observing that this term tends to 0.

On the other hand, by definition, $F(i) \leq F^{\Psi}(i)$, so we arrive at equality. □

Corollary 3.1. *If D holds or N holds under Assumption A, then an optimal policy exists.*

Neither Theorem 3.6 nor Corollary 3.1 are true for P (see Example 40).

Optimal Stopping Over a Finite Horizon

One way that the total expected cost can be finite is if it is possible to enter a state from which no further costs are incurred.

Suppose u can take on two possible values: $u = 0$ (stop) and $u = 1$ (continue). Suppose there is a termination (absorbing) state, say 0, that is entered upon choosing the stopping action. Once this state is entered the system stays in that state and no further cost is incurred. We let $c(x,0) = k(x)$ (stopping cost) and $c(x,1) = c(x)$ (continuation cost).

Suppose that $F_s(x)$ denotes the minimum total cost when we are constrained to stop within the next s steps. This gives a finite-horizon problem with dynamic programming equation

$$F_s(x) = \min\{k(x), c(x) + \mathbb{E}[F_{s-1}(X_1)|X_0 = x, u_0 = 1]\} \tag{3.33}$$

with $F_0(x) = k(x), c(0) = 0$.

Consider now the set of states in which it is at least as good to stop now as to continue one more step, and then stop:

$$\mathscr{L} = \{x : k(x) \leq c(x) + \mathbb{E}[k(X_1)|X_0 = x, u_0 = 1]\}.$$

It is obviously not optimal to stop if $x \notin \mathscr{L}$. If $\mathscr{L}$ is closed then the following theorem gives us the form of the optimal policies for all finite-horizons.

Theorem 3.7 (OSLA)**.** *Suppose $\mathscr{L}$ is closed (so that once the system enters $\mathscr{L}$ it remains in $\mathscr{L}$). Then an optimal policy for all finite horizons is: stop iff $x \in \mathscr{L}$.*

Proof. This proof is by induction. If the horizon is $s = 1$, then obviously it is optimal to stop only if $x \in \mathscr{L}$. Suppose the theorem is true for a horizon of $s-1$. As above, if $x \notin \mathscr{L}$, then it is better to continue for one more step and stop rather than stop in state x. If $x \in \mathscr{L}$, then the fact that $\mathscr{L}$ is closed implies $x_1 \in \mathscr{L}$ and so $F_{s-1}(x_1) = k(x_1)$. But then (3.33) gives $F_s(x) = k(x)$. So we should stop if $x \in \mathscr{L}$. □

The optimal policy is known as the one-step look-ahead rule (OSLA rule).

***Example* 43.** Optimal parking. Recall Problem 2 from Lecture 1. Let $x = 1$ denote the car park is available x spaces from the entrance, and $x = 0$ denote that it is full. We have for $s > 0$

$$\begin{aligned} F_s(0) &= (1-p)F_{s-1}(0) + pF_{s-1}(1), \\ F_s(1) &= \min\{s, (1-p)F_{s-1}(0) + pF_{s-1}(1)\} \end{aligned}$$

where $F_0(0) = D$ and $F_0(1) = 0$.

Now, consider the set $\mathscr{L}$ for this problem. We have

$$\mathscr{L} = \{s : s \le e(s-1)\}$$

where $e(s)$ is the expected cost if we take the first available parking space that is s spaces or closer to the entrance. (Note that "taking the first available parking space is equivalent to stopping at the first opportunity.") Then,

$$e(s) = ps + (1-p)e(s-1),$$

with $e(0) = (1-p)D$. Solving, we have

$$e(s) = \frac{-(1-p)}{p} + s + \left(D + \frac{1}{p}\right)(1-p)^{s+1}, \ s = 0, 1, \ldots.$$

Simplifying, we have

$$\mathscr{L} = \{s : (Dp+1)(1-p)^s \geq 1\}.$$

This set is closed, as s is decreasing, and hence Theorem 3.7 holds and $\mathscr{L}$ characterises the optimal policy.

Optimal Stopping Over an Infinite Horizon

Let $F_s(x)$ be the infimal cost given that we are required to stop by the sth step. Let $F(x)$ be the equivalent where the only requirement is that we eventually stop. Since less cost can be incurred if we are allowed more time in which to stop, we have

$$F_s(x) \geq F_{s+1}(x) \geq F(x).$$

Hence, by monotone convergence $F_s(x)$ tends to a limit, say $F_\infty(x)$, and $F_\infty(x) \geq F(x)$.

The following Lemma gives conditions under which the infimal finite-horizon cost does converge to the infimal infinite-horizon cost.

Lemma 3.1. *Suppose all costs are bounded as follows.*

$$K = \sup_x k(x) < \infty, \text{ and } C = \inf_x c(x) > 0. \tag{3.34}$$

Then $F_s(x) \to F(x)$ as $s \to \infty$.

Proof. Suppose π is an optimal policy for the infinite horizon problem and stops at the random time τ. Then its cost is at least $(s+1)C\Pr(\tau > s)$. However, since it would be possible to stop at time 0 the cost is also no more than K, so

$$(s+1)C\Pr(\tau > s) \leq F(x) \leq K.$$

In the $s-$horizon problem we could follow π, but stop at time s if $\tau > s$. This implies

$$F(x) \le F_s(x) \le_* F(x) + K\Pr(\tau > s) \le F(x) + \frac{K^2}{(s+1)C}.$$

By letting $s \to \infty$, we have $F_\infty(x) = F(x)$.

Note that the inequality $\le_*$ comes considering $F_s(x)$ conditioning on whether $\tau \le s$ or $\tau > s$. □

Note that the problem posed here is identical to one in which we pay K at the start and receive a terminal reward $r(x) = K - k(x)$.

Theorem 3.8. *Suppose $\mathscr{L}$ is closed and 3.34 hold. Then an optimal policy for the infinite horizon is: stop iff $x \in \mathscr{L}$.*

Proof. By Theorem 3.7 we have for all finite s,

$$F_s(x) \begin{cases} = k(x), & x \in \mathscr{L} \\ < k(x), & x \notin \mathscr{L} \end{cases}$$

Lemma 3.1 gives $F(x) = F_\infty(x)$. □

Chapter 4

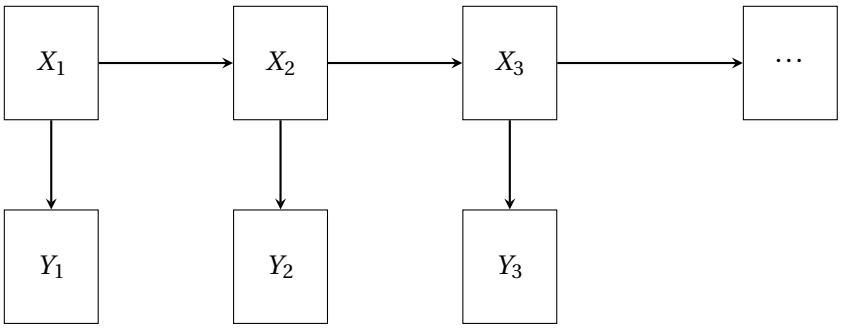
Hidden Markov Chains

§4.1 Introduction

A *hidden Markov chain* is a stochastic process composed of two discrete-time stochastic processes, one of which is the *state sequence* and the other is the *observation sequence*.

The observation sequence is the observed sequence of data/symbols, and the state sequence is the unobserved sequence we wish to infer.

In a hidden Markov chain, the state sequence is a discrete-time Markov chain, $\{X_n, n \geq 1\}$, which is assumed to be unobserved. The observation sequence $\{Y_n, n \geq 1\}$ is typically not Markovian, Y_n is a function of X_n, and the observation sequence is conditionally independent given the state sequence.



The joint stochastic process $\{(X_n, Y_n), n \geq 1\}$ is Markovian. We have the following definition of a hidden Markov chain.

Defn 4.1. *A hidden Markov chain (HMC) is a Markovian collection of random variables* $\{X_n, Y_n, n \geq 1\}$ *such that*

$$\Pr\left(\boldsymbol{Y}_t = \boldsymbol{y}_k | \boldsymbol{X}_t = \boldsymbol{x}_t\right) = \prod_{i=1}^{t} \Pr(Y_i = y_i | X_i = x_i) \tag{4.1}$$

where

$$\boldsymbol{Y}_t := (Y_0, Y_1, \cdots, Y_t), \quad \boldsymbol{X}_t := (X_0, X_1, \cdots, Y_t),$$
$$\boldsymbol{y}_t := (y_0, y_1, \cdots, y_t), \quad \boldsymbol{x}_t := (x_0, x_1, \cdots, x_t).$$

The condition (4.1) is the conditional independence property of the observation random variables.

PARAMETERISATION

Here we introduce the necessary components in order to study hidden Markov chains. We have that the joint p.m.f. (we assume that the observation se-

quence is also discrete in this course) is

$$\begin{aligned}
&\Pr\left(\boldsymbol{X}_t = \boldsymbol{x}_t, \boldsymbol{Y}_t = \boldsymbol{y}_t\right) \\
&= \Pr\left(\boldsymbol{X}_t = \boldsymbol{x}_t\right) \Pr\left(\boldsymbol{Y}_t = \boldsymbol{y}_t | \boldsymbol{X}_t = \boldsymbol{x}_t\right) \\
&= \underbrace{\Pr(X_0 = x_0) \prod_{i=1}^{t} \Pr(X_i = x_i | X_{i-1} = x_{i-1})}\overbrace{\prod_{j=0}^{t} \Pr(Y_j = y_j | X_j = x_j)},
\end{aligned}$$

where the first product (underbraced) is based on the Markov property, and the second product (overbraced) is based on the conditional independence property of the observation random variables $\{Y_n\}$.

Notation

We have that the

$$\Pr(X_t = j | X_{t-1} = i) =: p_{ij}$$

and are collated in the transition probability matrix P, assumed to be of size $N \times N$. Associated with this transition matrix are parameters, which we denote by a vector $\boldsymbol{\theta}_P$.

The observation p.m.f. $\Pr(Y_t = y | X_t = x)$ will be, in general, denoted as $p(y|x)$. It will have parameters which we denote by vector $\boldsymbol{\phi}_p$.

Finally, we have the initial distribution $\Pr(X_0 = i)$, which we denote $p_0(i)$ and its associated vector by $\boldsymbol{p_0}$.

***Example* 44.** Suppose you would like to determine a measure of the average rainfall in Adelaide during 1960-1965. Your answer should be either WET or DRY over each year. Unfortunately, there is are no rainfall records during this year.

You notice that another recent study of the particular location of interest has revealed that currently the probability of a WET year being followed by a WET year is 0.6. Similarly the probability that a DRY year is followed by a DRY year is 0.75. The same study concluded that this process of average rainfall

can reasonably be expected to have been fairly consistent with the distant past.

Hence, we have a DTMC model of the evolution of the average rainfall measure given by the probability transition matrix

$$\begin{array}{c} \\ \text{WET} \\ \text{DRY} \end{array} \begin{array}{c} \begin{array}{cc} \text{WET} & \text{DRY} \end{array} \\ \begin{pmatrix} 0.60 & 0.40 \\ 0.25 & 0.75 \end{pmatrix} \end{array} \tag{4.2}$$

There is a known correlation between the size of growth rings of trees and the average rainfall per season at this particular location involving its own species of trees. This gives us an observable data sequence from the older trees that existed in the distant past at the location of interest.

The size of the growth rings is coarsely classified as either small (s), medium (m), or large (ℓ). On available evidence, there is a probabilistic relationship between average annual rainfall and the tree ring growth size given by the following

$$\begin{array}{c} \\ \text{WET} \\ \text{DRY} \end{array} \begin{array}{c} \begin{array}{ccc} s & m & \ell \end{array} \\ \begin{pmatrix} 0.10 & 0.30 & 0.60 \\ 0.70 & 0.25 & 0.05 \end{pmatrix} \end{array} \tag{4.3}$$

Although we cannot observe the average yearly rainfall from the distant past, we do have a record of the growth rings from the trees that existed at that time. This record and our model built on (4.2) and (4.3) will give us the possible **state sequence** of average yearly rainfall even though it is not directly observable.

Note that we use

$$p_{ij} = \Pr(X_t = j \mid X_{t-1} = i)$$

for the homogeneous DTMC probability transitions that describe the evolution of the **state sequence**. In general, it is an $N \times N$ matrix of transition probabilities across a set of N states.

In this particular example, $N = 2$ and the state space $\mathscr{S} = \{W, D\}$. The parameters $\boldsymbol{\theta}_p$ are:

$$p_{WW} = 0.60, \quad p_{WD} = 0.40, \quad p_{DW} = 0.25, \quad p_{DD} = 0.75.$$

Also, we have a $N = 2 \times 3 = M$ matrix of probabilities

$$\Pr(\text{observation } y_t \text{ at time } t \mid \text{state } x_t \text{ at time } t).$$

The possible observations are $\{s, m, \ell\}$ and so the parameters $\boldsymbol{\phi}_p$ are

$$\begin{aligned} \Pr(s \mid W) &= 0.10, \quad \Pr(s \mid D) = 0.70, \\ \Pr(m \mid W) &= 0.30, \quad \Pr(m \mid D) = 0.25, \\ \Pr(\ell \mid W) &= 0.60, \quad \Pr(\ell \mid D) = 0.05, \end{aligned}$$

Finally, we have some initial state distribution $\Pr(X_0 = i)$, which we denote by $p_0(i) : \boldsymbol{p}_0 = \big(p_0(W), p_0(D)\big)$ in the tree ring model. In particular we assume that

$$\boldsymbol{p}_0 = (0.40, 0.60).$$

In general, we have a model that is comprised of the parameters $\Lambda = \big(\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0\big)$ and a sequence of observations $\boldsymbol{y}$.

THREE CLASSICAL PROBLEMS

Three classical problems associated with HMC

1. Given $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$ and $\boldsymbol{y}$, we find $\Pr(\boldsymbol{y} \mid \boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$. That is, we find the likelihood of $\boldsymbol{y}$ under the given model, or, equivalently, evaluating the marginal probability of observation $\boldsymbol{y}$.

2. Given $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$ and $\boldsymbol{y}$, we find the most likely state sequence for the underlying Markov process. In a sense, we want to uncover the hidden part of the HMM.

3. Given $\boldsymbol{y}$ and the dimensions of $(\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$, we find a model $(\boldsymbol{\theta}'_p, \boldsymbol{\phi}'_p, \boldsymbol{p}'_0)$ that maximises the probability of observing $\boldsymbol{y}$. Here, we estimate the parameters of the model using maximum likelihood.

§4.2 CLASSIC PROBLEM ONE

Consider the first of our problems: given an observation sequence of a hidden Markov chain with known parameters $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p_0})$, evaluate the marginal probability of an observation sequence $\boldsymbol{y}$.

A naïve approach to this problem is to use the direct definition, by marginalising the joint density:

$$\begin{aligned}\Pr(\boldsymbol{Y}_T = \boldsymbol{y}_T) &= \sum_{x_1=1}^{N}\sum_{x_2=1}^{N}\cdots\sum_{x_k=1}^{N}\Pr(X_0 = x_0)\prod_{i=1}^{T}\Pr(X_i = x_i \mid X_{i-1} = x_{i-1}) \\ &\quad\times\left(\prod_{i=0}^{T}\Pr(Y_i = y_i \mid X_i = x_i)\right) \\ &= \sum_{x_1=1}^{N}\sum_{x_2=1}^{N}\cdots\sum_{x_k=1}^{N} p_0(x_0)\prod_{i=1}^{T} p_{x_{i-1},x_i}\left(\prod_{i=0}^{T} p(y_i|x_i)\right). \qquad (4.4)\end{aligned}$$

***Example* 45.** Returning to our rainfall example, suppose we have the observation $\boldsymbol{y} = (s, m, s, \ell)$. If $\boldsymbol{x} = (W, W, D, D)$, then

$$\begin{aligned}
&\Pr((W,W,D,D), \boldsymbol{y} \mid (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)) \\
&= \Pr(X_0 = W)\Pr(X_1 = W|X_0 = W)\Pr(X_2 = D|X_1 = W)\Pr(X_3 = D|X_2 = D) \\
&\times \Pr(Y_0 = s|X_0 = W)\Pr(Y_1 = m|X_1 = W)\Pr(Y_2 = s|X_2 = D)\Pr(Y_3 = \ell|X_3 = D) \\
&= 0.0000756.
\end{aligned}$$

Note that each of the probability terms on the RHS of the first equation is conditional on $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$; for notational simplicity we suppressed the conditioning on Λ. Evaluating the joint probabilities of the given $\boldsymbol{y}$ and each possible state sequence $\boldsymbol{x}$, we have

$$\begin{aligned}
&\Pr((W,W,W,W), \boldsymbol{y} \mid \Lambda) = 0.00015552, && \Pr((W,W,W,D), \boldsymbol{y} \mid \Lambda) = 0.00000864, \\
&\Pr((W,W,D,W), \boldsymbol{y} \mid \Lambda) = 0.00030240, && \Pr((W,D,W,W), \boldsymbol{y} \mid \Lambda) = 0.00003600, \\
&\Pr((D,W,W,W), \boldsymbol{y} \mid \Lambda) = 0.00068040, && \Pr((W,W,D,D), \boldsymbol{y} \mid \Lambda) = 0.00007560, \\
&\Pr((W,D,W,D), \boldsymbol{y} \mid \Lambda) = 0.00000200, && \Pr((W,D,D,W), \boldsymbol{y} \mid \Lambda) = 0.00031500, \\
&\Pr((D,W,W,D), \boldsymbol{y} \mid \Lambda) = 0.00003780, && \Pr((D,W,D,W), \boldsymbol{y} \mid \Lambda) = 0.00132300, \\
&\Pr((D,D,W,W), \boldsymbol{y} \mid \Lambda) = 0.00070875, && \Pr((W,D,D,D), \boldsymbol{y} \mid \Lambda) = 0.00007875, \\
&\Pr((D,W,D,D), \boldsymbol{y} \mid \Lambda) = 0.00033075, && \Pr((D,D,W,D), \boldsymbol{y} \mid \Lambda) = 0.00003938, \\
&\Pr((D,D,D,W), \boldsymbol{y} \mid \Lambda) = 0.00620156, && \Pr((D,D,D,D), \boldsymbol{y} \mid \Lambda) = 0.00155039,
\end{aligned}$$

So, the marginal probability of $\boldsymbol{y}$ is

$$\Pr(\boldsymbol{y} \mid \Lambda) = \sum_{\boldsymbol{x}} \Pr(\boldsymbol{x}, \boldsymbol{y} \mid \Lambda) = 0.01184594.$$

However, in most real-world problems of interest the naïve approach will be excessively computationally intensive (and infeasible). The number of multiplications required is $(2T+1)N^{T+1}$, where N^{T+1} is the number of possible sequences $\boldsymbol{x} = (x_0, \ldots, x_T)$.

We now consider an alternative approach.

Defn 4.2. *Define the* ***forward function*** *as*

$$\alpha_t(i \mid \Lambda) = \Pr(\boldsymbol{Y}_t = \boldsymbol{y}_t, X_t = i \mid \Lambda) \tag{4.5}$$

for $i = 1, \ldots, N$ *and* $t = 0, \ldots, T$.

Defn 4.3. *The* ***forward algorithm*** *("*α*-pass") is*

Step 1. For $i = 1, 2, \ldots, N$, *let*

$$\begin{aligned} \alpha_0(i \mid \Lambda) &= \Pr(y_0, X_0 = i \mid \Lambda) \\ &= \Pr(X_0 = i \mid \Lambda) \Pr(y_0 \mid X_0 = i, \Lambda). \end{aligned} \tag{4.6}$$

Step 2. For $t = 1, 2, \ldots, T$ *and* $i = 1, 2, \ldots, N$, *compute*

$$\alpha_t(i \mid \Lambda) = \left(\sum_{j=1}^{N} \alpha_{t-1}(j \mid \Lambda)\, p_{ji} \right) \Pr(y_t \mid X_t = i, \Lambda). \tag{4.7}$$

Step 3. Then, the probability of an observation sequence $\boldsymbol{y}_T$ *given* Λ *is given by*

$$\Pr(\boldsymbol{y}_T \mid \Lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

The proof of the equality in the second step is as follows.

Proof. We have

$$
\begin{aligned}
a_t(i) &= \Pr\left(\boldsymbol{Y}_t = \boldsymbol{y}_t, X_t = i\right) \\
&= \sum_{j=1}^{N} \Pr\left(\boldsymbol{Y}_t = \boldsymbol{y}_t, X_t = i, X_{t-1} = j\right) \\
&= \sum_{j=1}^{N} \Pr\left(Y_t = y_t, X_t = i \mid \boldsymbol{Y}_{t-1} = \boldsymbol{y}_{t-1}, X_{t-1} = j\right) \Pr\left(\boldsymbol{Y}_{t-1} = \boldsymbol{y}_{t-1}, X_{t-1} = j\right) \\
&= \sum_{j=1}^{N} \Pr\left(y_t \mid \boldsymbol{y}_{t-1}, X_{t-1} = j, X_t = i\right) \Pr\left(X_t = i \mid \boldsymbol{y}_{t-1}, X_{t-1} = j\right) \Pr\left(\boldsymbol{y}_{t-1}, X_{t-1} = j\right) \\
&=^{*} \Pr\left(y_t \mid X_t = i\right) \sum_{j=1}^{N} \Pr\left(X_t = i \mid X_{t-1} = j\right) \Pr\left(\boldsymbol{y}_{t-1}, X_{t-1} = i\right) \\
&= \left(\sum_{j=1}^{N} \alpha_{t-1}(j)\, p_{ji}\right) \Pr(y_t \mid X_t = i).
\end{aligned}
$$

Note that for $=^{*}$, we have used the conditional independence property of the hidden Markov chain, and the Markov property of $\{X_t\}$. □

Now consider the evaluation of the forward function using the trellis diagram of Figure 4.1.

We can see that to evaluate the first time segment (column 1 on trellis), we require N multiplications from (4.6). Now consider the evaluation of $\alpha_1(1)$. This requires $N+1$ multiplications from (4.7), and such a procedure needs to be carried out for each $\alpha_1(i)$ for $i = 1,\ldots,N$, resulting in $N(N+1)$ multiplications to evaluate the second time segment. Then, the third time segment is similarly evaluated in $N(N+1)$ multiplications, and so on through time. Hence, the total number of multiplications required to completely evaluate the trellis is $N+N(N+1)T$ and the total number of additions required is $N(N-1)T$.

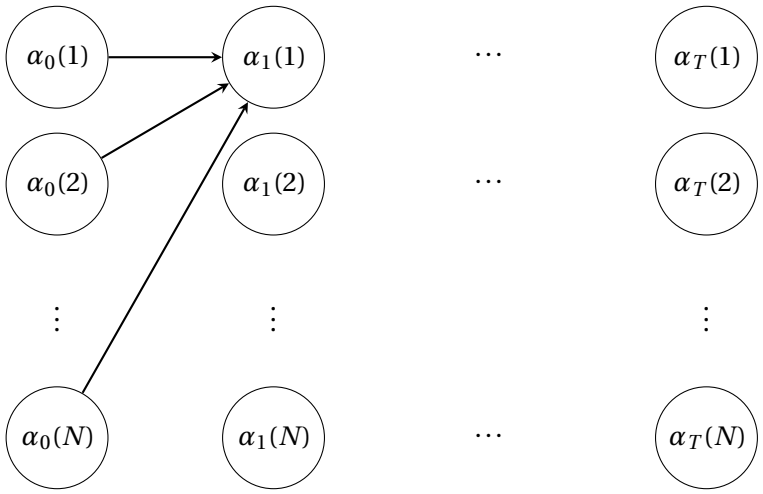


Figure 4.1: *Trellis diagram of the forward algorithm*

§4.3 Classic Problem Two

Given $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$ and a sequence of observations $\boldsymbol{y}_T = (y_0, \ldots, y_T)$, we want to calculate

(a) the most likely state x_t at time t:

$$\widehat{x}_t = \arg\max_{x_t} \Pr(X_t = x_t \mid \boldsymbol{y}_T, \Lambda);$$

(b) the most likely state sequence $\boldsymbol{x}_T = \{x_0, \ldots, x_T\}$:

$$\boldsymbol{x}_T = \arg\max_{\boldsymbol{x}_t} \Pr(\boldsymbol{X}_T = \boldsymbol{x}_T \mid \boldsymbol{y}_T, \Lambda).$$

First, to calculate the most likely state x_t at time t, we use the **forward-backward algorithm.**

Defn 4.4. *Define the* ***backward function*** *as*

$$\beta_t(i \mid \Lambda) = \Pr((y_{t+1}, y_{t+2}, \ldots, y_T) \mid X_t = i, \Lambda) \tag{4.8}$$

for $i = 1, \ldots, N$ *and* $t = 0, \ldots, T$.

That is, $\beta_t(i)$ is the probability of the partial observation sequence from time t until time T, conditioned on the state sequence being in state i at time t. We can also calculate $\beta_t(i)$ recursively, using the backward algorithm.

Defn 4.5. *The* ***backward algorithm*** *("β-pass") is*

Step 1. Let $\beta_T(i \mid \Lambda) = 1$ *for* $i = 1, 2, \ldots, N$.

Step 2. For $t = T-1, T-2, \ldots, 0$ *and* $i = 1, 2, \ldots, N$, *compute*

$$\beta_t(i) = \sum_{j=1}^{N} p_{ij} \Pr(y_{t+1} \mid X_{t+1} = j, \Lambda) \beta_{t+1}(j). \tag{4.9}$$

The proof of (4.9) is presented in class.

Defn 4.6. *The **forward-backward algorithm** is:*
for $t = 0, 1, \ldots, T-1$ and for $i = 1, 2, \ldots, N$, define

$$\begin{aligned} \gamma_t(i) &:= \Pr(X_t = i \mid \boldsymbol{y}, \Lambda) \\ &= \frac{\alpha_t(i)\beta_t(i)}{\Pr(\boldsymbol{y} \mid \Lambda)}. \end{aligned} \tag{4.10}$$

Thus,the most likely state at time t, given the observation sequence $\boldsymbol{y}$, is the state for which $\gamma_t(i)$ is maximum over the index of states $i \in \mathscr{S}$.

Intuitively, Equality (4.10) comes from the fact that $\alpha_t(i)$ measures the relevant probability up to time t and $\beta_t(i)$ measures the relevant probability after t.

Considering the trellis diagram for the backward algorithm in Figure 4.2, the total number of multiplications required to evaluate all $\beta_t(i)$ is $2N^2T$ and the total number of additions is $N(N-1)T$.

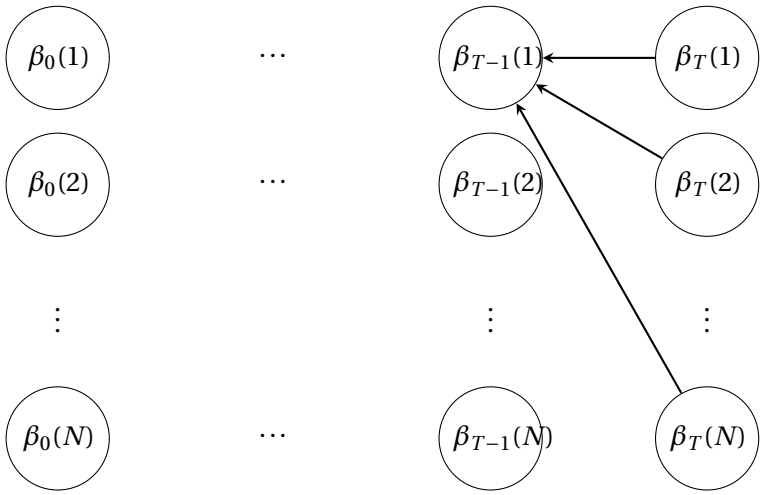


Figure 4.2: *Trellis diagram of the backward algorithm*

To determine the most likely state sequence $\boldsymbol{x}_T$, we use the Viterbi algorithm. This dynamic programming approach is basically an α-pass with the sum operation being replaced by the max operation.

Defn 4.7 (The Viterbi algorithm)**.**

1. *For* $i = 1,2,\ldots,N$, *let*

$$\delta_0(i) = p_0(i)\Pr(y_0 \mid X_0 = i).$$

2. *For* $t = 1,2,\ldots,T$ *and* $i = 1,2,\ldots,N$, *compute*

$$\delta_t(i) = \max_{j=1,2,\ldots,N}\left(\delta_{t-1}(j)\,p_{ji}\Pr(y_t \mid X_t = i)\right).$$

At each successive t, the Viterbi algorithm finds the probability of the best path ending at each of the states $i = 1, 2, \ldots, N$.

***Example* 46.** Recall our tree ring model with observation sequence (s, m, s, ℓ). We have

$$\delta_0(W) = \Pr(X_0 = W, y_0 = s) = p_0(W)\Pr(y_0 = s \mid X_0 = W) = 0.4(0.1) = 0.04,$$
$$\delta_0(D) = \Pr(X_0 = D, y_0 = s) = p_0(D)\Pr(y_0 = s \mid X_0 = D) = 0.6(0.7) = 0.42.$$

Hence, for $t = 1$,

$$\begin{aligned}\delta_t(W) &= \max_{W,D}\{\delta_0(W)p_{WW}\Pr(y_1 = m \mid X_1 = W), \delta_0(D)p_{DW}\Pr(y_1 = m \mid X_1 = W)\} \\ &= \max_{W,D}\{0.04(0.6)(0.3),\ 0.42(0.25)(0.3)\} \\ &= \max_{W,D}\{0.0072, 0.0315\} = 0.0315.\end{aligned}$$

So the most likely path of length 2 ending with W is DW, which has probability 0.0315.

Similarly,

$$\begin{aligned}\delta_t(D) &= \max_{W,D}\{\delta_1(W)p_{WD}\Pr(y_1 = m \mid X_1 = D), \delta_1(D)p_{DD}\Pr(y_1 = m \mid X_1 = D)\} \\ &= \max_{W,D}\{0.42(0.4)(0.25),\ 0.42(0.75)(0.25)\} \\ &= \max_{W,D}\{0.004, 0.0788\} = 0.0788.\end{aligned}$$

So the most likely path of length 2 ending with D is DD, which has probability 0.0788. We can continue this process one stage at a time for as long as we have an observation to get the diagram depicted in Figure 4.3.

The arrows point to the predecessor in the optimal path from the particular ending state.

Note that at each stage, using the Viterbi algorithm, we only need to maintain a minimal list of the highest scoring paths at each possible stage,

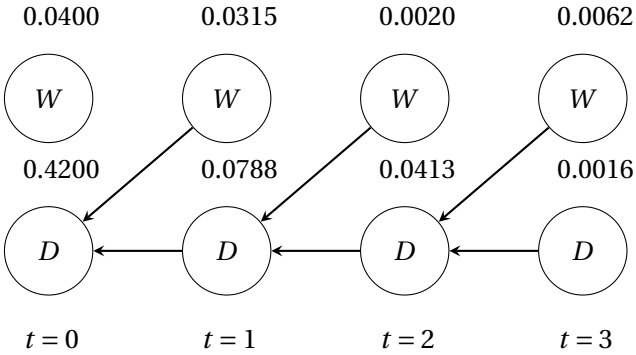


Figure 4.3: *Most probable path*

which is the key to all of the DP algorithm efficiencies. The path of length 4 with maximum probability 0.0062 of occurrence ends in state W and tracking back we get the path (D,D,D,W), just as we would see in our brute force approach.

SCALING

As the probabilities get progressively smaller, there is a problem of potential underflow which we might address by taking logarithms to preserve the integrity of our calculations. For example, we could rewrite the Viterbi algorithm in terms of logarithms and once again keep a list of predecessors to to get the optimal path.

Defn 4.8 (The log Viterbi algorithm)**.**

1. *For* $i = 1, 2, \ldots, N$, *let*

$$\widehat{\delta}_0(i) = \log\big(p_0(i)\big) + \log\Big(\Pr(y_0 \mid X_0 = i)\Big).$$

2. *For* $t = 1, 2, \ldots, T$ *and* $i = 1, 2, \ldots, N$, *compute* $\widehat{\delta}_t(i)$ *given by*

$$\max_{j=1,2,\ldots,N}\Big[\widehat{\delta}_{t-1}(j) + \log\big(p_{ji}\big) + \log\Big(\Pr(y_t \mid X_t = i)\Big)\Big].$$

At each successive t, the log Viterbi algorithm finds the best path of highest probability ending at each state $i = 1, 2, \ldots, N$, by considering $\max_{j=1,\ldots,N}\big(\widehat{\delta}_t(j)\big)$. The smaller the probability the more negative the log value.

***Example* 47.** Recall again our tree ring model with observation sequence (s, m, s, ℓ), where we see that

$$\begin{aligned}
\widehat{\delta}_0(W) &= \log\big(p_0(W)\big) + \log\big(\Pr(y_0 = s \mid X_0 = W)\big) \\
&\approx -0.9163 - 2.30260 = -3.2189, \\
\widehat{\delta}_0(D) &= \log\big(p_0(D)\big) + \log\big(\Pr(y_0 = s \mid X_0 = D)\big) \\
&\approx -0.5108 - 0.3567 = -0.8675.
\end{aligned}$$

Then,

$$\widehat{\delta}_0(W) = -3.2189 < \widehat{\delta}_0(D) = -0.8675,$$

yielding the same result we saw before.

The arrows point to the predecessor in the optimal path from the particular ending state: $e^{-5.0832} = 0.0062$.

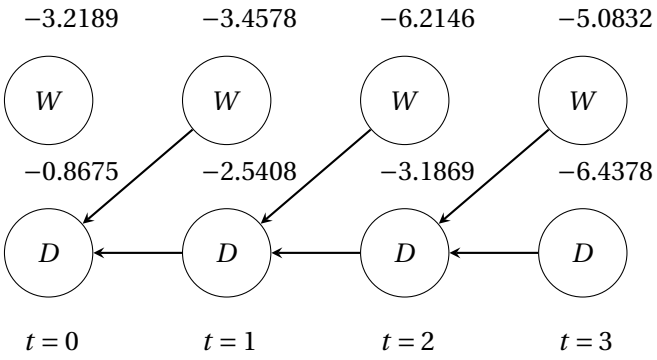


Figure 4.4: *Most probable path in log terms*

§4.4 Classical Problem Three

In the Classical Problem Three, we want to estimate the maximum likelihood estimate of parameters given an observation sequence $\boldsymbol{y}_T$. The values of N, the number of possible values for the state sequence $\{X_t\}$, and of M, the number of possible values for the observation sequence $\{Y_t\}$, are fixed. The values of $\boldsymbol{p}_0$ and the elements of the probability matrices – the elements of $(\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$ — are to be determined.

To this end, we use the **Baum-Welch algorithm**, which is a special case of the **Expectation-Maximisation (EM) algorithm**. The latter is used for finding MLEs of model parameters where there is incomplete or missing data. In this case, the 'hidden' realised state sequences are the missing data. Note that the Baum-Welch algorithm finds a local maximum, and not necessarily a global maximum, for

$$(\boldsymbol{\theta}_p^*, \boldsymbol{\phi}_p^*, \boldsymbol{p}_0^*) = \arg \max_{\{(\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)\}} \Pr(\boldsymbol{y} \mid \boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0).$$

For $t = 0, 1, \ldots, T-1$ and $i, j \in \{1, 2, \ldots, N\}$, define

$$\begin{aligned}\xi_t(i,j) &= \Pr(X_t = i, X_{t+1} = j \mid \boldsymbol{y}, \Lambda) \\ &= \frac{\Pr(X_t = i, X_{t+1} = j, \boldsymbol{y} \mid \Lambda)}{\Pr(\boldsymbol{y} \mid \Lambda)} \\ &= \frac{\alpha_t(i) p_{ij} \Pr(y_{t+1} \mid X_{t+1} = j, \Lambda) \beta_{t+1}(j)}{\Pr(\boldsymbol{y} \mid \Lambda)}. \end{aligned} \tag{4.11}$$

The $\gamma_t(i)$ and $\xi_t(i,j)$ are related by

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j). \tag{4.12}$$

Defn 4.9. *Given $\gamma_t(i)$ and $\xi_t(i,j)$, we can update the model parameters as follows.*

STEP 1 *For $i = 1, 2, \ldots, N$, let*

$$p_0(i) = \gamma_0(i).$$

STEP 2 *For $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, N$ we compute*

$$p_{ij} = \sum_{t=0}^{T-1} \xi_t(i,j) \Bigg/ \sum_{t=0}^{T-1} \gamma_t(i).$$

STEP 3 *For $j = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, M$ we compute*

$$\Pr(k \mid j) = \sum_{\substack{t \in \{0,1,\ldots,T-1\} \\ y_t = k}} \gamma_t(j) \Bigg/ \sum_{t=0}^{T-1} \gamma_t(j).$$

The numerator on the RHS of the expression at Step 2 of Definition 4.9 gives the expected number of transitions from state i to state j, conditioned

on the observation $\boldsymbol{y}$. That is, let N_{ij} be the number of transitions from state i to state j over the time period $[0, T-1]$, and $I_t(i,j)$ be the indicator function that of the event $\{X_t = i, X_{t+1} = j\}$, then

$$\begin{aligned}
\mathbb{E}\left[N_{ij} \mid \boldsymbol{y}, \Lambda\right] &= \mathbb{E}\left[\sum_{t=0}^{T-1} I_t(i,j) \mid \boldsymbol{y}, \Lambda\right] \\
&= \sum_{t=0}^{T-1} \mathbb{E}\left[I_t(i,j) \mid \boldsymbol{y}, \Lambda\right] \\
&= \sum_{t=0}^{T-1} P(X_t = i, X_{t+1} = j \mid \boldsymbol{y}, \Lambda) \\
&= \sum_{t=0}^{T-1} \xi_t(i,j).
\end{aligned}$$

Similarly, the denominator on the RHS of the expression at Step 2 is the expected number of transitions from state i to any other state:

$$\begin{aligned}
\mathbb{E}\left[\sum_{j=1}^{N} N_{ij} \mid \boldsymbol{y}, \Lambda\right] &= \mathbb{E}\left[\sum_{j=1}^{N}\sum_{t=0}^{T-1} I_t(i,j) \mid \boldsymbol{y}, \Lambda\right] \\
&= \sum_{j=1}^{N}\sum_{t=0}^{T-1} P(X_t = i, X_{t+1} = j \mid \boldsymbol{y}, \Lambda) \\
&= \sum_{t=0}^{T-1}\sum_{j=1}^{N} P(X_t = i, X_{t+1} = j \mid \boldsymbol{y}, \Lambda) \\
&= \sum_{t=0}^{T-1} \gamma_t(i).
\end{aligned}$$

Similarly, the numerator on the RHS of the expression at Step 3 of Definition 4.9 is the expected number of times that the process is in state j with observation k, while the denominator is the expected number of times that the process is in state j. The ratio again is then clearly a re-estimation (or an update) of $\Pr(k \mid j)$.

Re-estimation is an iterative process where we first initialise $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$ with our best estimate, or we can just choose random variables such that

$$p_0(i) \approx \frac{1}{N}, \quad p_{ij} \approx \frac{1}{N}, \quad \Pr(k \mid j) \approx \frac{1}{M}. \tag{4.13}$$

Note that if the exact uniform distribution is used for the initiation, it will result in a local maximum from which the estimation procedure cannot escape.

The solution to Classical Problem Three can be summarised as

Step 1. **Initialise:** $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$.

Step 2. **Compute:** $\alpha_t(i), \beta_t(i), \gamma_t(i)$ and $\xi_t(i,j)$.

Step 3. **Re-estimate:** $\Lambda' = (\boldsymbol{\theta}'_p, \boldsymbol{\phi}'_p, \boldsymbol{p}'_0)$.

Step 4. **If** $\Pr(\boldsymbol{y} \mid \Lambda)$ increases*,
GOTO Step 4.4
else
we stop the iterations.
end

---

*Stopping when $\Pr(\boldsymbol{y} \mid \Lambda)$ does not significantly increase is generally used.

Exercise: Suppose we have two dices, one loaded and one fair. The observation sequence we have is 53536511211152211. Estimate the parameters Λ using the Baum-Welch algorithm.

SCALING

Another scaling technique is as follows. When computing the forward function $\alpha_t(i)$, we can normalise each $\alpha_t(i)$ by dividing by $\sum_{j=1}^{N} \alpha_t(j)$. That is, for $i = 1, \ldots, N$ and $t = 0, \ldots, T$, we evaluate the conditional probabilities

$$\widehat{\alpha}_t(i) = \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)} = \Pr(X(t) = i \mid y_0, \ldots, y_t, \Lambda). \tag{4.14}$$

Defn 4.10. *An algorithm for evaluating $\widehat{\alpha}_T(i)$ is as follows.*

1. *For $t = 0$ and for $i = 1, 2, \ldots, N$, let*

$$\widetilde{\alpha}_0(i) = \alpha_0(i), \quad C_0 = \frac{1}{\sum_{j=1}^{N} \widetilde{\alpha}_0(j)},$$
$$\widehat{\alpha}_0(i) = C_0 \widetilde{\alpha}_0(i).$$

2. *For $t = 1, \ldots, T$ we set*

$$\widetilde{\alpha}_t(i) = \left(\sum_{j=1}^{N} \widehat{\alpha}_{t-1}(j) p_{ji} \right) \Pr(y_t \mid X_t = i),$$
$$C_t = \frac{1}{\sum_{j=1}^{N} \widetilde{\alpha}_t(j)},$$
$$\widehat{\alpha}_t(i) = C_t \widetilde{\alpha}_t(i).$$

We need to check that the re-estimation formulae holds. That is, we need to show that $\widehat{\alpha}_t(i)$ evaluated by the above algorithm is indeed (4.14).

Proof. Since $\widehat{\alpha}_0(i) = C_0 \widetilde{\alpha}_0(i) = C_0 \alpha_0(i)$, we assume that

$$\widehat{\alpha}_t(i) = C_0 C_1 \ldots C_t \alpha_t(i),$$

then

$$\begin{aligned}
\widehat{\alpha}_{t+1}(i) = C_{t+1} \widetilde{\alpha}_{t+1}(i) &= C_{t+1} \left(\sum_{j=1}^{N} \widehat{\alpha}_t(j) p_{ji} \right) \Pr(y_{t+1} \mid X_{t+1} = i) \\
&= C_0 C_1 \ldots C_{t+1} \left(\sum_{j=1}^{N} \alpha_t(j) p_{ji} \right) \Pr(y_{t+1} \mid X_{t+1} = i) \\
&= C_0 C_1 \ldots C_{t+1} \alpha_{t+1}(i).
\end{aligned}$$

Hence, our assumption $\widehat{\alpha}_t(i) = C_0 C_1 \dots C_t \alpha_t(i)$ holds for all t by the principle of mathematical induction. Then, by the definitions of $\widetilde{\alpha}$ and $\widehat{\alpha}$ it follows that

$$\widehat{\alpha}_t(i) = \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)} \tag{4.15}$$

and hence the $\widehat{\alpha}_t(i)$ are the desired scaled versions of $\alpha_t(i)$ for all t. □

Equation (4.15) implies that for $t = 0, 1, \dots, T$

$$\sum_{j=1}^{N} \widehat{\alpha}_t(j) = 1,$$

and so by our assumption

$$\begin{aligned} \sum_{j=1}^{N} \widehat{\alpha}_T(j) &= C_0 C_1 \dots C_{t+1} \sum_{j=1}^{N} \alpha_T(j) \\ &= C_0 C_1 \dots C_{t+1} \Pr(\boldsymbol{y} \mid \Lambda). \end{aligned} \tag{4.16}$$

It can be shown that

$$\Pr(\boldsymbol{y} \mid \Lambda) = \frac{1}{\prod_{j=1}^{T} C_j},$$

and to avoid underflow we calculate

$$\log\big(\Pr(\boldsymbol{y} \mid \Lambda)\big) = -\sum_{j=1}^{T} \log\big(C_j\big). \tag{4.17}$$

This same scale factor method can be similarly shown to hold for calculation of the $\beta_t(i)$ so that $\widehat{\beta}_t(i) = C_t \beta_t(i)$. We can then calculate the $\gamma_t(i)$ and $\xi_t(i, j)$ using the previous formulae except now we use the $\widehat{\alpha}_t(i)$ and $\widehat{\beta}_t(i)$ instead of $\alpha_t(i)$ and $\beta_t(i)$. The resultant $\gamma_t(i)$ and $\xi_t(i, j)$ can then be used to re-estimate the model parameters $\Lambda = (\boldsymbol{\theta}_p, \boldsymbol{\phi}_p, \boldsymbol{p}_0)$ in the same way as before.

THE EM ALGORITHM

The Expectation-Maximisation algorithm is as follows. Let $\boldsymbol{y}$ be the observed data vector, $\boldsymbol{x}$ be the unobserved data vector, and $\boldsymbol{\theta}$ be the set of parameters to estimate. Our goal is to find the MLE

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \underbrace{\log f(\boldsymbol{y} \mid \theta)}_{\text{incomplete-data log-likelihood}}.$$

Defn 4.11. *Expectation-Maximisation Algorithm.*

STEP 1. *Set* $n = 0$ *and choose a starting point* $\boldsymbol{\theta}_0$.

STEP 2. EXPECTATION. *Calculate*

$$\mathbb{E}_{\boldsymbol{\theta}_n}[\log f(\boldsymbol{y}, \boldsymbol{x} \mid \boldsymbol{\theta}) \mid \boldsymbol{y}] =: Q(\boldsymbol{\theta}, \boldsymbol{\theta}_n),$$

where $f(\boldsymbol{y}, \boldsymbol{x} \mid \boldsymbol{\theta})$ *is the complete-data log-likelihood.*

STEP 3. MAXIMIZATION. *Set* $n = n+1$, *let*

$$\boldsymbol{\theta}_n := \arg\max_{\boldsymbol{\theta} \in \Theta} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_{n-1}).$$

If $\boldsymbol{\theta}_n \neq \boldsymbol{\theta}_{n-1}$, *return to Step* 2.