Linear programming algorithms for lower previsions

Nawapon Nakharutai

A Thesis presented for the degree of
PhD in Mathematical Sciences

Statistics and Probability
Department of Mathematical Sciences
Durham University
England

March 2019

Linear programming algorithms for lower previsions

Nawapon Nakharutai

Submitted for the Doctor of Philosophy degree of

PhD in Mathematical Sciences

March 2019

Abstract

The thesis begins with a brief summary of linear programming, three methods for solving linear programs (the simplex, the affine scaling and the primal-dual methods) and a brief review of desirability and lower previsions. The first contribution is to improve these algorithms for efficiently solving these linear programming problems for checking avoiding sure loss. To exploit these linear programs, I can reduce their size and propose novel improvements, namely, extra stopping criteria and direct ways to calculate feasible starting points in almost all cases. To benchmark the improvements, I present algorithms for generating random sets of desirable gambles that either avoid or do not avoid sure loss.

Overall, the affine scaling and primal-dual methods benefit from the improvements, and they both outperform the simplex method in most scenarios. Hence, I conclude that the simplex method is not a good choice for checking avoiding sure loss. If problems are small, then there is no tangible difference in performance between all methods. For large problems, the improved primal-dual method performs at least three times faster than any of the other methods.

The second contribution is to study checking avoiding sure loss for sets of desirable gambles derived from betting odds. Specifically, in the UK betting market, bookmakers usually provide odds and give a free coupon, which can be spent on

betting, to customers who first bet with them. I investigate whether a customer can exploit these odds and the free coupon in order to make a sure gain, and if that is possible, how can that be achieved. To answer this question, I view these odds and the free coupon as a set of desirable gambles and present an algorithm to check whether and how such a set incurs sure loss. I show that the Choquet integral and complementary slackness can be used to answer these questions. This can inform the customers how much should be placed on each bet in order to make a sure gain.

As an illustration, I show an example using actual betting odds in the market where all sets of desirable gambles derived from those odds avoid sure loss. However, with a free coupon, there are some combinations of bets that the customers could place in order to make a guaranteed gain.

I also consider maximality which is a criterion for decision making under uncertainty, using lower previsions. I study two existing algorithms, one proposed by Troffaes and Hable (2014), and one by Jansen, Augustin, and Schollmeyer (2017). For the last contribution in the thesis, I present a new algorithm for finding maximal gambles and provide a new method for generating random decision problems to benchmark these algorithms on generated sets.

To find all maximal gambles, Jansen et al. solve one large linear program for each gamble, while in Troffaes and Hable, and also in our new algorithm, this can be done by solving a larger sequence of smaller linear programs. For the second case, I apply efficient ways to find a common feasible starting point for this sequence of linear programs from the first contribution. Exploiting these feasible starting points, I propose early stopping criteria for further improving efficiency for the primal-dual method.

For benchmarking, we can generate sets of gambles with pre-specified ratios of maximal and interval dominant gambles. I investigate the use of interval dominance at the beginning to eliminate non-maximal gambles. I find that this can make the problem smaller and benefits Jansen et al.'s algorithm, but perhaps surprisingly,

not the other two algorithms. We find that our algorithm, without using interval dominance, outperforms all other algorithms in all scenarios in our benchmarking.

Declaration

The work in this thesis is based on research carried out at the Statistics and Probability Group, the Department of Mathematical Sciences, Durham University, England. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referred to the contrary in the text. Part II is based on my work under a supervision of Matthias Troffaes and Camila Caiado. In particular, chapters 7, 8 and 9 closely follow [26, 27], chapter 10 closely follows [29] and chapters 11 and 12 closely follow [28].

Dedicated to

Papa

Acknowledgements

This thesis has been funded by the Development and Promotion of Science and Technology Talents Project (the Royal Government of Thailand scholarship).

This thesis would not have been completed without the guidance and help of several wonderful people. However, some deserve special mention.

First of all, I would like to express my sincere gratitude and indebtedness to my supervisors, Dr. Matthias Troffaes and Dr Camila Caiado, for their guidance, encouragement and support throughout my research. Thanks also to my former supervisor, Dr. Parkpoom Phetpradap, for his advice and support when I was in Thailand.

Besides my supervisors, I wish to thank my colleagues, especially James McRedmond, Tim Whitbread, Themistoklis Botsas and Clare Wallace for English grammar checks and nice discussions that never relate to any research. Thanks also to Rob Little, Daniel Ballesteros-Chavez and Wanchalerm Promduang who work in the fish-tank and usually come to my office for snacks, and along with Joseph Farrow, Calum Robson, Dan Rutter and Junbin Chen for lunchtime in the department. For little chats in the morning, I have to thank Pamela, Anne and Gail for teaching me to be a proper Geordie, but unfortunately, they failed.

I would like to thank my housemates: Ting He, Nipada Santha and Apichayaporn Ratkata for the experiences that we have been through together in the house. I do not think I could survive in that house without them.

Also thanks to my close friends, Wanwipha Sriwirat and Chakkrapong Kuensaen,

for remotely supporting me and making an effort to meet up with me when I was in Thailand. You two remind me of the song “Born To Be Your Friend”.

Penultimately, I would like to thank my family, particularly, papa, mama and my lovely (and sometimes naughty) sister, for their love, support, and understanding even though they have no clue what I do for my research. Without them, my work would not be possible.

Finally, I would like to thank my boyfriend, Supachai Awiphan, for your love, understanding and supporting me, and putting up with my complaints during the daily calls that we make. Thank you for being there for me since I started applying for the funding until the present, and hopefully for the future as well.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Contributions and outline

Consider a situation where we want to model uncertainty about an experiment. We usually build a statistical model and make a decision under uncertainty based on information that we have. If a complete probability measure on the state of nature can be specified, then we can simply model uncertainty through probability measures. However, we often face difficulties such as incomplete information or a contradiction between expert opinions. Therefore, we may not be able to specify a complete probability measure. In that case, we might consider other ways to express our beliefs.

One way to deal with such difficulties is expressing our beliefs using *gambles*. A gamble represents an uncertain reward (for instance, monetary) that depends on the outcomes of an event. To express our beliefs about the outcomes of the experiment, we can simply state a collection of gambles that we are willing to accept. A set of gambles that we accept is called a *set of desirable gambles*, which can handle partial information in a more robust way.

We can also model uncertainty about an experiment via acceptable buying (or

selling) prices for gambles called *lower previsions*. The study of lower previsions can be traced back to several early works , for example, Boole [3], de Finetti [5] and Smith [36]. Based on these works, Williams [48] generalised the idea of previsions and gave rationality criteria for lower previsions based on axioms of acceptability. The concept of lower previsions in his work can be seen as lower bounds on expectation. The theory of lower previsions was developed further by Walley [46].

Since we can choose any gamble to be desirable, there may be some combinations of desirable gambles that result in a certain loss. In this case, our set of desirable gambles is not reasonable. To prevent this situation, Williams [48, 49] gave a full axiomatic treatment for sets of desirable gambles, and formalized a consistency principle called *avoiding sure loss*. This concept can be applied to lower previsions as well. We will briefly review desirability and lower previsions in chapter 6.

We can verify whether a set of desirable gambles avoids sure loss by solving a linear programming problem [46, p.151] (see section 7.1), which was further studied and extended by various authors [30, 47]. In the literature, to the best of our knowledge, there has been little to no discussion about which algorithm should be used to solve linear programming problems for avoiding sure loss. Walley [46, p. 511] suggested that *Karmarkar's* method may be useful for solving large problems. However, these days Karmarkar's method is considered obsolete in favour of other interior point methods, e.g. the affine scaling and the primal-dual methods [1].

Among several linear programming algorithms, the simplex method is one of the easiest and most commonly used. The affine scaling method is an improved version of Karmarkar's method. The primal-dual method is arguably the best interior-point algorithm. To understand more about these methods, in chapter 3, we study the simplex method, in chapter 4, we look at the affine scaling, and in chapter 5, we study the primal-dual method.

The first contribution in this thesis is a comparative study and analysis of how

we can solve linear programs for checking avoiding sure loss most effectively by these three methods. To do so, in chapter 7, we first exploit the structure of this linear program and its dual. We slightly reduce the size of the linear program and propose two improvements to the three methods, namely, an extra stopping criterion to detect unboundedness and a direct way to calculate the feasible starting points, which therefore can reduce the effort required in the pre-solve phase of some of these algorithms. We also discuss in more detail both the advantages and disadvantages of each method for checking avoiding sure loss. For benchmarking these improvements, in chapter 8, we provide algorithms that generate random sets of desirable gambles that either avoid or do not avoid sure loss and compare the impact of our improvements on these three methods. We present results and discussion in chapter 9.

Next, we move to another contribution which is presented in chapter 10, where we study sets of desirable gambles derived from betting odds and free coupons, and then check avoiding sure loss. Specifically, in the UK betting market, bookmakers usually provide odds and give a free coupon, which can be spent on betting, to customers who first bet with them. To predict the outcome of a match, the bookmaker may face some difficulties as mentioned before, for example, a lack of data (e.g. Manchester United has never played with Leeds United during last five years), limited football expert opinion, or even contradicting information from different football experts. These issues can also be handled by using sets of desirable gambles. In this case, the bookmaker can model his belief about this outcome by stating a collection of gambles that he is willing to offer.

A bookmaker's set of desirable gambles avoids sure loss if there are no combinations of desirable gambles that result in a sure loss. Therefore, there is no combination of bets from which customers can make a guaranteed gain. In contrast, if the set does not avoid sure loss, then there is a combination of bets that customers can exploit to incur a sure gain.

In addition to avoiding sure loss, the bookmakers also want to entice new customers. There are several techniques that bookmakers can use to encourage customers to bet with them. Some bookmakers may offer greater betting odds than others as greater odds imply a greater payoff to the customers. Another technique is to offer a *free coupon* which is a stake that customers can spend on betting. We view the free coupon as a part of a desirable gamble.

Naturally, bookmakers do not want customers to bet on a combination of different odds and free coupons that incurs a guaranteed profit. Therefore, from the bookmaker's point of view, they would like to check whether sets of desirable gambles derived from different odds and free coupons avoid sure loss or not. On the other hand, in theory, a customer may be interested in the case where the bookmaker's set of desirable gambles does not avoid sure loss, because then the customer can make a guaranteed profit. In that case, a customer may want to find a strategy of bets which results in the best possible sure gain.

Several authors study how customers can exploit betting odds and free bets in order to find strategies that can make a profit. For example, Walley [46, Appendix I] and Quaeghebeur et al. [31] study applications of sets of desirable gambles on sports; Milliner et al. [23], Schervish et al. [35], Vlastakis et al. [45] directly exploit betting odds, whilst Emiliano [6] takes free bets into account. Emiliano [6] studies the case of only two possible outcomes, and allows cooperation between customers against bookmakers. In this thesis, we generalise it to any finite number of possible outcomes, but we only consider a single customer. We view betting odds and free coupons as a set of desirable gambles and check whether such a set avoids sure loss or not. If the set does not avoid sure loss, then we show exactly how a customer can bet in order to make a sure gain.

For this specific problem, instead of solving a linear programming problem for checking avoiding sure loss, we can check avoiding sure loss via the *natural extension*

which is a tool for deductive inference in desirability [24, p. 40]. In this case, we can compute the natural extension through the *Choquet integral* or through solving a linear program where the optimal value is equal to the natural extension. In the case of not avoiding sure loss, we show that we can obtain a strategy that the customer can bet on to make a guaranteed gain by using the Choquet integral and *complementary slackness* conditions. We also find optimal solutions of the corresponding pair of dual linear programming programs without directly solving them. As an illustration, we show that for some actual betting odds in the market, all sets of desirable gambles derived from those odds avoid sure loss. However, with a free coupon, there are some combinations of bets that the customer could make in order to make a guaranteed gain.

Finally, we turn our attention to decision making with lower previsions. Suppose that a subject must choose from a set of possible decisions, where each decision leads to an uncertain reward, depending on her decision and on the state of nature revealed after the decision. Again, the reward could be anything, for example, money, food, or a lottery ticket. For simplicity, we assume that rewards are expressed on a utility scale. In this way, we can view an uncertain reward, and thereby, each decision, as a gamble.

Naturally, the subject would like to choose gambles that lead to the best possible reward. Given a set of gambles, the subject may have some criteria to eliminate some gambles in the set, and the set of all remaining gambles is called an *optimal* set.

Maximality [46, §3.9.1-3.9.3, pp. 160-162] and *interval dominance* [51, §2.3.3, pp. 68-69] are well-known decision criteria induced by partial orders corresponding to lower previsions [38]. Several authors, for example, Jansen et al. [13], Kikuti et al. [17] and Troffaes and Hable [40, p. 336], propose algorithms for finding maximal gambles. Specifically, Jansen et al. [13] propose an algorithm that can verify whether

a gamble is maximal by solving a single linear program. In contrast, to check maximality of each gamble by algorithms proposed in Kikuti et al. [17] and Troffaes and Hable [40, p. 336], one has to evaluate the sign of the lower prevision of several gambles. This implies that we have to solve several smaller linear programming problems [40, p. 331].

Troffaes and Hable [40, Algorithm 16.4, p. 336] present an incremental algorithm where once some maximal gambles in the sets are identified, we should compare the remaining gambles against those maximal gambles first. Additionally, Troffaes and Hable [40, p. 336] suggest that sorting all gambles in advance, e.g. by expectation, might help the algorithm to perform better. Incorporating this suggestion, we present a new algorithm for finding maximal gambles and benchmark these different approaches by conducting an extensive simulation study.

Linking to the first contribution, we use the primal-dual method, which is particularly suitable when working with lower previsions as it can quickly obtain feasible starting points, to solve these linear programs. In addition, exploiting the fact that the primal-dual method solves both the primal and dual simultaneously, we propose early stopping criteria to determine more quickly the sign of lower previsions.

Since all maximal gambles are also interval dominant, if a gamble is not interval dominant, then it is also not maximal [38]. So, Troffaes [38] suggested to apply interval dominance first in order to eliminate non-maximal gambles. In the case that most gambles are not interval dominant, this can eliminate many non-maximal gambles early on. Therefore, in addition to compare the above mentioned algorithms for finding maximal gambles, we perform these algorithms with and without applying interval dominance at the beginning.

The contributions of improving and benchmarking algorithms for finding maximal gambles are as follows. We propose a new algorithm for finding maximal gambles in chapter 11 and compare its performance to the two algorithms proposed

by Jansen et al. [13] and Troffaes and Hable [40, p. 336] in chapter 12. For the algorithm in Troffaes and Hable [40, p. 336], and also for our new algorithm, we solve a sequence of linear programs by the primal-dual method, as we can easily obtain feasible starting points and early stopping criteria can be implemented in the method. For benchmarking, we present an algorithm for generating sets of gambles which have a precisely predetermined number of maximal and interval dominant gambles. We perform these algorithms on generated sets and then present results and discussion.

Finally, chapter 13 concludes the thesis. An overview of the structure of the thesis is given in the diagram in fig. 1.1.

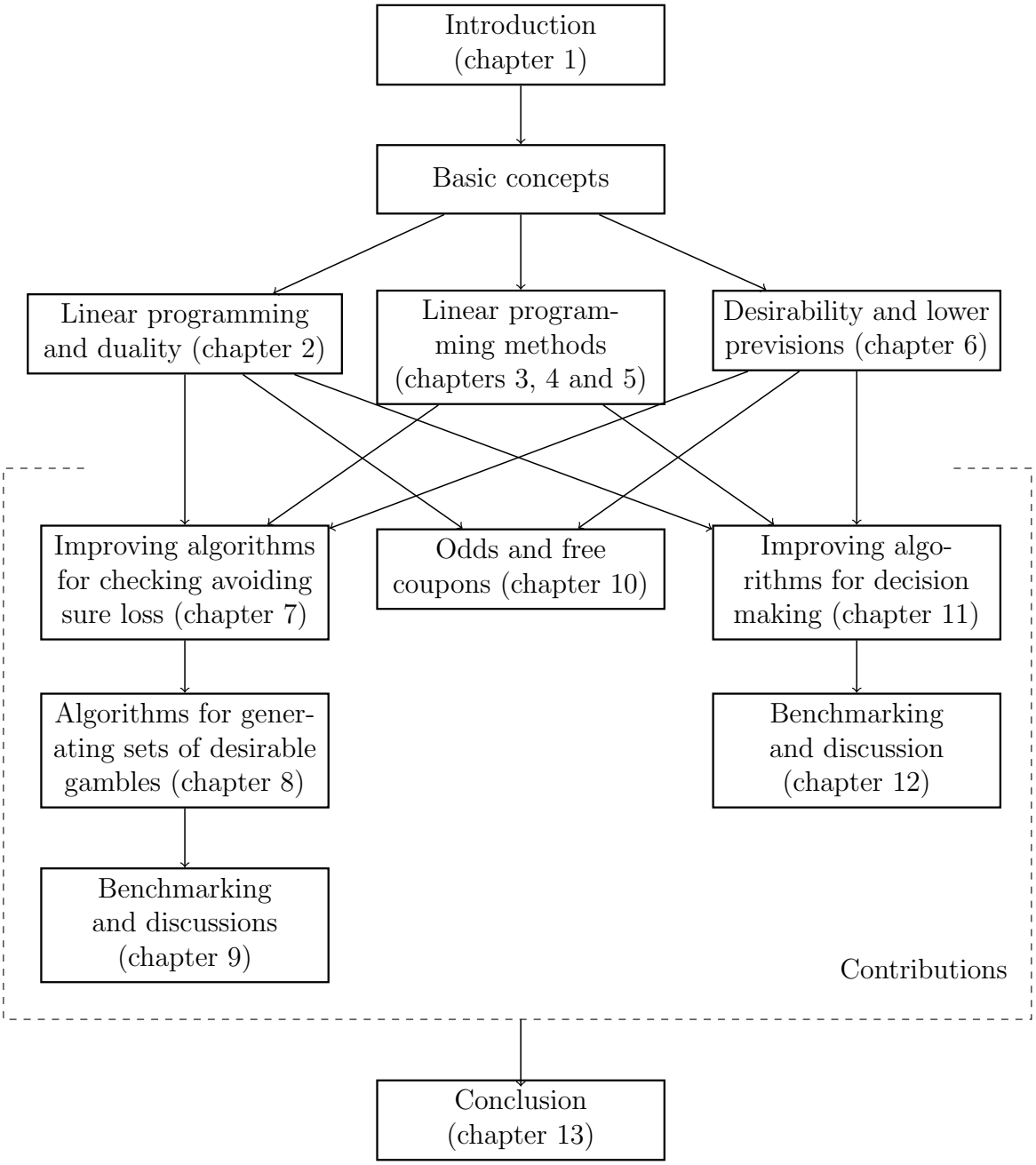

Figure 1.1: Flow diagram: the structure of the thesis.

Part I

Basic concepts

Chapter 2

Linear programming and duality

As we mentioned before, one way to check avoiding sure loss is to solve a linear programming problem. To study linear programming problems for checking avoiding sure loss in chapter 7, we need to understand linear programming problems and how to solve them.

In this chapter, we first explain linear programming problems and study some relations between primal and dual problems including duality theory and complementary slackness. These relations will be used when we study two interior point methods: the affine scaling and the primal-dual methods in chapters 4 and 5. We will also use complementary slackness again in chapter 10 where we study betting odds and free coupons.

2.1 Elements of linear programming problems

A linear programming problem is a problem of optimising a linear function subject to constraints of linear equalities and linear inequalities. Since maximising $c^\top x$ is equivalent to minimising $-c^\top x$, it is sufficient to consider only minimising the linear

function. We can write every problem in standard form as in eq. (P):

$$\min c^\top x \text{ subject to } Ax = b,\ x \geq 0, \tag{P}$$

where $A \in \mathbb{R}^{m\times n}$ with rank m and $m \leq n$, $x, c \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$. Note that if a constraint is not in the from of equality, for example,

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq b_1, \tag{2.1}$$

then we can add non-negative variable, s_1, which is called a *slack variable*, to make this constraint become an equality constraint:

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n + s_1 = b_1. \tag{2.2}$$

We assume that A has linearly independent rows (i.e. rank $A = m$). If not, say $A_3 = \lambda_1 A_1 + \lambda_2 A_2$ for $\lambda_1, \lambda_2 > 0$, then either $\lambda_1 b_1 + \lambda_2 b_2 = b_3$ and hence we remove the third row from A and b or $\lambda_1 b_1 + \lambda_2 b_2 \neq b_3$ and hence there is no solution for this linear programming problem.

Definition 1. A vector x is called a *feasible solution* if x satisfies $Ax = b$ and $x \geq 0$. □

The collection of all feasible solutions is called a feasible set, namely, $F = \{x : Ax = b, x \geq 0\}$. A linear programming problem is called *feasible* if there exists a feasible solution; otherwise, it is an *infeasible* problem.

Definition 2. The problem (P) is called *unbounded* if for all $\lambda \in \mathbb{R}$, there exists a feasible solution x such that $c^\top x \leq \lambda$. □

Definition 3. An *optimal solution* x^* is a feasible solution that achieves the optimal value of the objective function, and $c^\top x^*$ is called the *optimal value*. □

As we assume that A has linearly independent rows, we can define solutions to

a linear programming problem as follows.

Definition 4. A *basis* B is a collection of m of the n variables for which the corresponding m columns of A are linearly independent. If a variable is in the basis, then it is called a *basic* variable; otherwise it is called a *non-basic* variable. □

Definition 5. A *basic* solution is a vector x such that $Ax = b$ and all of the $n - m$ non-basic variables are zero. In addition, if $x \geq 0$, then x is called a *basic feasible* solution. □

Note that every constraint is either a linear equality or a linear inequality which forms a feasible half-space or plane. These feasible half-space and planes are convex. The feasible set which is the intersection of these convex sets is also convex.

Theorem 6. [33, p.40] Let $F = \{x : Ax = b, x \geq 0\}$. The point x is an extreme point of F if and only if x is a basic feasible solution of the problem (P). □

In other words, basic feasible solutions are precisely the extreme points of the feasible region. The fundamental theorem of linear programming is presented as follows:

Theorem 7. [20, p.24] If a linear programming problem has an optimal solution, then an optimal solution can be found at an extreme point. □

Definition 8. If a linear programming problem has basic feasible solutions with $n - m + 1$ or more zero elements, then the problem is called *degenerate*. If all basic feasible solutions are zero (this happens when $b = 0$), then the linear programming problem is called *fully degenerate*. □

In the case of fully degenerate problems, their feasible region are cones, and therefore have only one extreme point, namely, the origin. As we will see later chapter 7, one way to check avoiding sure loss is to solve a fully degenerate problem. The following lemma is useful for finding the optimal value of a fully degenerate problem.

Lemma 9. (A generalised version of [44, p. 42, exercise 3.4].) The linear programming problem

$$\min c^\top x \text{ subject to } Ax \geq 0 \tag{2.3}$$

either has optimal value equal to zero, or is unbounded. □

Proof. We will show that this linear program has an optimal value that is either zero or unbounded in two steps. We first show that zero is always a feasible value for the linear program. This means that the optimal value of this problem must be less or equal to zero. Next, we show that if zero is not optimal, then the problem must be unbounded. Consequently, we conclude that either the optimal value of this fully degenerate linear programming problem is zero or the problem is unbounded.

1. We see that $x = 0$ is a feasible solution of eq. (2.3). Therefore, $c^\top x = 0$ is a feasible value for this linear program.

2. Suppose the linear program is feasible for some x^* such that $c^\top x^* = \alpha < 0$. Consider any $\beta < 0$. Since $\frac{\beta}{\alpha} > 0$, we can multiply $Ax \geq 0$ with $\frac{\beta}{\alpha}$. So, $\frac{\beta}{\alpha}(Ax^*) = A(\frac{\beta}{\alpha}x^*) \geq 0$. So, this fully degenerate problem is feasible for $x = \frac{\beta}{\alpha}x^*$ with corresponding value β for the objective function. Since we can make β arbitrarily negative, the problem is unbounded.

□

Lemma 9 shows that, for fully degenerate problems, if there is a feasible solution x such that $c^\top x < 0$, then the problem is unbounded. Therefore, in our algorithms, we can stop early as soon as we find a negative value.

2.2 Duality theory

Duality is one of the most important concepts for solving linear programming problems. To start this section, we state a standard form of a dual problem. Then we study the fundamental relations between a pair of primal and dual problems. These relations include weak and strong duality theorems and complementary slackness conditions. These concepts are then used to derive several linear programming methods, for example, the dual simplex algorithm and interior point methods.

Every linear programming problem (P) has an associate linear programming problem as in eq. (D):

$$\max b^\top y \text{ subject to } A^\top y + t = c,\ t \geq 0,\ y \text{ free.} \tag{D}$$

We usually call (P) the *primal* problem and (D) the dual problem. This pair of problems are dual to each other, so we can choose either one of the pair as the primal problem and the other one becomes its dual problem.

If a pair of problems are feasible, then the primal objective value is always bounded below by the dual objective value:

Theorem 10. (Weak duality [33, p.70].) If x is a primal feasible solution and y is a dual feasible solution, then $c^\top x \geq b^\top y$. □

Consequently, we derive the following results from the weak duality theorem:

Corollary 11. [33, p.71]

1. Let x be (P)-feasible and let y be (D)-feasible. If $c^\top x = b^\top y$, then x is (P)-optimal and y is (D)-optimal.

2. If the (P) problem is unbounded, then the (D) problem does not have a feasible solution.

3. If the (D) problem is unbounded, then the (P) problem does not have a feasible solution.

□

A stronger result shows that the converse of corollary 11 also holds.

Theorem 12. (Strong duality [33, p.71].)

1. If either of (P) or (D) has a finite optimal solution, then so does the other, and their optimal values are equal.

2. Let (P) have a feasible solution. Then (D) has no feasible solution if and only if the objective function of the (P) problem is unbounded.

3. Let (D) have a feasible solution. Then (P) has no feasible solution if and only if the objective function of the (D) problem is unbounded.

□

2.3 Complementary slackness

We now look at a relationship between the variables of one linear program and the constraints of its dual. This relation can be used to find an optimal value of the variables given the optimal values of the slack variables of the other program.

Suppose that a pair of linear programs are in the form (P) and (D). Let x and y be feasible solutions of the pair of linear programs. As

$$Ax - b^\top = 0 \quad \text{and} \quad t = c - A^\top y, \tag{2.4}$$

we see that

$$c^\top x - b^\top y = (c^\top - y^\top A)x + y^\top(Ax - b^\top) = t^\top x. \tag{2.5}$$

The quantity $t^\top x$ is called as the *duality gap*. Note that the duality gap is zero, if, and only if, $t^\top x = 0$. This statement leads to the following theorem.

Theorem 13. (Complementary slackness [33, p.75].) Consider a pair of linear programs of the form (P) and (D). Let x be a primal feasible solution and let y be a dual feasible solution. Then x and y are optimal solutions of the pair if and only if they satisfy the *complementary slackness* condition:

$$x^\top(c - A^\top y) = x^\top t = 0. \tag{2.6}$$

□

Since x and t are non-negative, so eq. (2.6) is equivalent to $\forall j = 1, \ldots, n$: either x_j or t_j is equal to zero.

Given an optimal solution for (P) (or (D)), complementary slackness can be used to find an optimal solution of its dual problem [50, p. 329]. Note that we will use this technique later in chapter 10.

2.4 The development of methods to solve linear programs

There are many available methods for solve linear programming problems that are widely used in practice. The most well-known and simplest method is the simplex method which was proposed by Dantzig in 1949 [33, p.1]. The simplex method, which is an iterative method, performs row operations on the simplex table. At every iteration, the method moves from a current basic feasible solution to another basic feasible solution which improves the objective function value. The method

terminates when it cannot decrease the objective function value any more. Note that, in practical implementation, the revised simplex method is the most commonly used as it does not need to store the full simplex table. Therefore, the revised simplex method can save storage and computation.

However, when a linear programming problem is degenerate, that is, at least one basic variable is zero, the simplex algorithm may perform badly. Specifically, the simplex algorithm may *cycle* and it will not find an optimal solution [44, p.30]. Several attempts to prevent cycling are proposed. Two simple ways to avoid cycling are using *lexicographic* method or *Bland's rule* [7, §3.6].

Unfortunately, the simplex method has another issue called *stalling*, meaning that the method performs an exponentially long sequence of degenerate pivots. Klee and Minty [18] found a linear programming problem with n variables that has 2^n extreme points and the simplex method visits all the extreme points before obtaining an optimal solution. In that case, the simplex method needs $2^n - 1$ iterations. Indeed, the number of iterations may be exponential in the size of linear programs before obtaining an optimal solution. Although, the simplex method may stall under degeneracy, in practice, it is still one of the easiest and most commonly used algorithms.

In 1979, the first algorithm for linear programming problems that requires at most a polynomial number of iterations in the number of variables to obtain any optimal solution was introduced by Khachiyan [15]. It is called the *ellipsoid* method. Theoretically, the ellipsoid method was able to solve linear programming problems in a polynomial number of steps (as a function of the size of linear programs) in the worst case, but it turned out to be highly inefficient when it was implemented to solve general linear programming problems [20, p.112]. In 1984, Karmarkar [14] rediscovered a polynomial time algorithm, which was first discovered by Dikin in 1967, for the linear programming problem. This method is called the interior point

method. The idea of this method is to generate a sequence of interior feasible points which eventually converges to an optimal solution through the interior region [33, p.111]. When this method was implemented, it was very competitive with the simplex method, especially for large problems [34, p.273]. After that, several interior-point methods for solving linear programming problems have been developed, for example, projective methods, affine scaling methods and primal-dual methods (see [32] for a list of references). In this thesis, we consider only two interior point methods: the affine scaling and the primal-dual methods.

2.5 Summary

To summarise, we studied linear programming problems and important relations of duality. We also reviewed the development of linear programming algorithms. For each of the next three chapters, we will study each of the following methods: the simplex, the affine scaling and the primal-dual, for solving linear programs for checking avoiding sure loss.

Chapter 3

Simplex methods

The first linear programming algorithm that we are going to present is the simplex method. It is the easiest and the most widely used method for solving linear programming problems.

In this chapter, we first study the simplex method in section 3.1 and present a summary of the revised simplex method in section 3.2 which is an efficient implementation for the simplex method. As the simplex method needs a basic feasible solution to start with, we explain a technique to obtain such a solution in section 3.3. Finally, in section 3.4, we describe how the simplex method can avoid cycling when it solves degenerate problems.

3.1 Formulating simplex methods

Consider a linear programming problem of the form:

$$\text{(P)} \quad \min c^\top x \text{ subject to } Ax = b \text{ and } x \geq 0. \tag{3.1}$$

Let x^0 be an initial basic feasible solution. We can permute and relabel variables in x^0 as

$$x^0 = [x_B^0 \; x_N^0]^\top \tag{3.2}$$

where $x_B^0 \geq 0$ is the vector of basic variables and $x_N^0 = 0$ is the vector of non-basic variables. Then, the constraints and the objective function can be rewritten corresponding to the basic feasible solution as

$$A = [B \; N] \quad \text{and} \quad c = [c_B \; c_N] \tag{3.3}$$

where B is an $m \times m$ non-singular matrix and N is an $m \times (n-m)$ matrix. Given B, elements in every feasible solution x can be reordered as

$$x = [x_B \; x_N]^\top \text{ where } x_B \geq 0, \; x_N \geq 0. \tag{3.4}$$

Then, we can rewrite (P) as

$$\begin{aligned} \text{(Q)} \qquad \min \qquad & z \coloneqq c_B^\top x_B + c_N^\top x_N && \text{(Q1)} \\ \text{subject to} \qquad & Bx_B + Nx_N = b && \text{(Q2)} \\ & x_B \geq 0, \; x_N \geq 0. && \text{(Q3)} \end{aligned}$$

The original linear programming problem corresponding to the simplex tableau is presented in table 3.1.

	x_B	x_N	value
z	$-c_B^\top$	$-c_N^\top$	0
x_B	B	N	b

Table 3.1: The form of the initial simplex tableau.

As B is invertible, we can solve eq. (Q2) for x_B:

$$x_B = B^{-1}b - B^{-1}Nx_N. \tag{3.5}$$

If $x_N = 0$, then $x = [B^{-1}b\ 0]^\top$ is a basic solution. In addition, if $B^{-1}b \geq 0$, then x is a basic feasible solution. Substituting eq. (3.5) into eq. (Q1) results in

$$z = c_B^\top x_B + c_N^\top x_N = c_B^\top (B^{-1}b - B^{-1}Nx_N) + c_N^\top x_N \tag{3.6}$$

$$= c_B^\top B^{-1}b - (c_B^\top B^{-1}N - c_N^\top)x_N. \tag{3.7}$$

By eq. (3.6) and eq. (3.7), we can express the (Q) problem as a matrix equation:

$$\begin{bmatrix} 1 & 0 & c_B^\top B^{-1}N - c_N^\top \\ 0 & I & B^{-1}N \end{bmatrix} \begin{bmatrix} z \\ x_B \\ x_N \end{bmatrix} = \begin{bmatrix} c_B^\top B^{-1}b \\ B^{-1}b \end{bmatrix}, \tag{3.8}$$

where I is the identity matrix. Note that eq. (3.8) can be represented in the *simplex tableau* as in table 3.2.

	x_B	x_N	value
z	0	$c_B^\top B^{-1}N - c_N^\top$	$c_B^\top B^{-1}b$
x_B	I	$B^{-1}N$	$B^{-1}b$

Table 3.2: The form of simplex tableau in the current basis.

Suppose that x^0 is a current basic feasible solution, then the objective value z^0 is $c_B^\top B^{-1}b$ in eq. (3.7), as $x_B^0 = B^{-1}b \geq 0$ and $x_N^0 = 0$. Consequently, eq. (3.7) becomes

$$z = z^0 - (c_B^\top B^{-1}N - c_N^\top)x \tag{3.9}$$

where x is any feasible solution. Note that we call the vector $c_B^\top B^{-1}N - c_N^\top$ the *reduced costs*. If any of the reduced costs is negative, then the corresponding component of x_N^0 can be increased to obtain a reduction of the objective value z. This is

equivalent to bringing a non-basic variable into the basis. Consequently, if all of the reduced costs are non-negative, the current basic feasible solution x is an optimal solution [7, p. 31].

In the case that the current basic feasible solution is not optimal, we have to move to another basic feasible solution. In other words, we find a new basis. To do so, we push one variable out of the current basis and then pick a single non-basic variable into the basis.

Suppose that there exists a non-basic variable x_ℓ that has a negative reduced cost, so we will bring x_ℓ into the basis. In this case, the constraints m rows of equations eq. (3.8) can be rewritten as

$$x_i + a_i x_\ell = (B^{-1}b)_i \tag{3.10}$$

for each variable x_i. This means that increasing x_ℓ will change the values of some or all of the basic variables; meanwhile, we must take one basic variable out of the basis. We rewrite eq. (3.10) as follows:

$$x_i = (B^{-1}b)_i - a_i x_\ell. \tag{3.11}$$

We observe that

- if $a_i > 0$, then x_i decreases as x_ℓ increases and $x_i = 0$ when $x_\ell = \dfrac{(B^{-1}b)_i}{a_i}$.
- if $a_i = 0$ or $a_i < 0$, then we can not decrease x_i.

Therefore, the only basic variables that can leave the basis are those with $a_i > 0$. We can increase x_ℓ as long as the remaining variables remain non-negative. To do so, we find an index s that satisfies

$$\frac{(B^{-1}b)_s}{a_s} = \min_{1 \le i \le m} \left\{ \frac{(B^{-1}b)_i}{a_i} : a_i > 0 \right\}, \tag{3.12}$$

and then assign the value of x_ℓ as $\frac{(B^{-1}b)_s}{a_s}$ and choose x_s to leave the basis. This process is called the *minimal ratio test* [10, p. 131]. Once the previous basic variable x_s is pushed out and a new basic variable x_ℓ is assigned the value, we restore the form of eq. (3.8). The process of changing the basis is also called *pivoting* [10, p. 131].

Note that the form of eq. (3.8) can be restored by row operations because it preserves the equality constraints of the original formulation. For $a_i > 0$, we modify the x_ℓ's column so that the only non-zero coefficient in that column is in row s and has value 1. By using row operations, we can keep the form of eq. (3.8) up to a permutation of columns. Specifically, a basic variable is the one whose column constructs the identity matrix, while the rest of variables are non-basic variables [20, p. 41].

In the case that a reduced cost $c_\ell < 0$ and the corresponding column has all elements $a_i \leq 0$ for every row i, then we cannot pivot on any row. This is because none of the basic variables will decrease in value as x_ℓ is increased from zero, and therefore x_ℓ can grow arbitrarily large. This results in an unbounded value of the objective function:

$$z = c_B^\top B^{-1} b - c_\ell^\top x_\ell \longrightarrow \infty. \tag{3.13}$$

So in this case, the linear programming problem is unbounded [10, p. 131].

Summarising all previous arguments, we outline the simplex algorithm as follows:

Step 1 (initialization): Given a basic feasible solution $x = [x_B \; x_N]$.

Step 2 (check for optimality): If all of the reduced costs are non-negative, then x is an optimal solution, and we are done. Otherwise, go to Step 3.

Step 3 (choose entering variable): If there exists a non-basic variable x_ℓ that has a negative reduced cost, then we choose x_ℓ to be an entering variable.

Step 4 (check unboundedness): If all elements in the reduced cost column are negative, then we stop here and the problem is unbounded. Otherwise, go to Step 5.

Step 5 (choosing leaving variable): Apply the minimal ratio test as in eq. (3.12) to pick one of the current basic variables, say x_s, to leave the basis.

Step 6 (update tableau): Restore the form of eq. (3.8) by row operations to obtain a new basic feasible solution. Go back to Step 2.

Note that row operation can be easily calculated in the simplex tableau which is suitable for small problems as we can compute it by hand. However, the simplex tableau may not be suitable for an implementation as we have to store the whole table. If there are m constraints and n variables, the simplex tableau needs an $(m+1)\times(n+1)$ array. In addition, the complexity is $\mathcal{O}(mn)$ arithmetical operations per every pivot. In the case that n is much larger than m, the standard simplex method may not be good for an implementation [10, p. 140].

To address this issue, we will study an implementation of the simplex approach called the *revised simplex* algorithm. This version needs less storage and less computation compare to the standard simplex method.

3.2 The revised simplex algorithm

In this section, we first study a procedure of the revised simplex algorithm and then we discuss a storage for this version. We also compare computational work for the revised simplex with the original version in the simplex tableau.

A procedure of the revised simplex algorithm is as follows [11, p. 208]:

Step 1 (initialisation): Let $x = [x_B\ x_N]$ be a basic feasible solution and let $I_B \coloneqq \{1, 2, \ldots, m\}$ be the index set of the basic variables.

Step 2 (calculating new feasible solution): Calculate *simplex multipliers* w by solving $B^\top w = c_B$ and the reduced cost $\tilde{c}_j = w^\top A_j - c_j$ for all $j \notin I_B$.

Step 3 (checking optimality): If $\tilde{c}_j \geq 0$ for all $j \notin I_B$, then we stop here and the current feasible solution is optimal. Otherwise, we move to Step 4.

Step 4 (choosing entering variable): Choose x_ℓ with $c_\ell < 0$ to be a basic variable, for some $\ell \notin I_B$, and update the current column $\tilde{A}_\ell$ by solving $B\tilde{A}_\ell = A_\ell$.

Step 5 (checking unboundedness): If $\tilde{A}_\ell \leq 0$, then we stop here and the problem is unbounded.

Step 6 (choosing leaving variable): Find an index k such that

$$\frac{(B^{-1}b)_k}{a_k} = \min_{1 \leq i \leq m} \left\{ \frac{(B^{-1}b)_i}{a_i} : a_i > 0 \right\}$$

where all elements a_i are in the column $\tilde{A}_\ell$.

Step 7 (update basis): Set

$$\begin{aligned} I_B &\longleftarrow I_B \cup \{\ell\} \setminus \{k\} \\ B &\longleftarrow [A_j]_{j \in I_B} \\ x_\ell &\longleftarrow \frac{(B^{-1}b)_k}{a_k} \\ x_j &\longleftarrow x_j - a_i x_\ell. \end{aligned}$$

Go back to Step 2.

The revised simplex requires two arrays of length m to store the values of x_B and $\tilde{A}_\ell$, an array of length $n - m$ to store the reduced costs $\tilde{c}_j$, and an $m \times m$ array to store B. Indeed, if B is a *sparse* matrix, then it can be compressed and requires less storage [10, p. 140].

Note that B^{-1} is implicitly calculated in Steps 2 and 4 which can be done by several methods, for example, the *Gaussian elimination*, or the *LU factorization*. These methods are efficient and usually implemented in computer packages. Therefore, the complexity per iteration is $\mathcal{O}(m^2)$ arithmetical operations [10, p. 141].

3.3 Finding initial basic feasible solutions

As we mentioned before, the simplex method needs an initial basic feasible solution to start with. In this section, we will discuss the *two-phase method* which is one of the commonly used methods for obtaining an initial basic feasible solution.

Without loss of generality, we can assume that $b \geq 0$ by multiplying the equality constraint by -1 if it is necessary. If the equality constraint is in the form:

$$Ax \leq b \quad \text{where} \quad x \geq 0, \tag{3.14}$$

then we can add a slack variable $s \geq 0$, so eq. (3.14) becomes:

$$Ax + s = b \quad \text{where} \quad x \geq 0,\ s \geq 0. \tag{S}$$

In this case, we immediately have a starting basic feasible solution by setting $x = 0$ and $s = b$. However, in many situations, the constraints are not in the form of eq. (S). For example, if constraints are in the form

$$-15x_1 + 10x_2 - x_3 \geq 7 \tag{3.15}$$

$$x_1 + x_2 + x_3 = 1, \tag{3.16}$$

then there are no longer slack variables to add as the initial basic variables. In such a case, we have to find a starting basic feasible solution. This task can be done by applying the two-phase method.

Suppose that we have constraints in the form of eq. (P) but not in the form of eq. (S). We can modify the constraints by adding an artificial variable w, where $w \in \mathbb{R}$, to the constraints. Consider the phase-I problem:

$$\text{(phase-I)} \quad \min \sum_{i=1}^{k} w_i \quad \text{s.t. } Ax + w = b,\ x \geq 0,\ w \geq 0. \tag{3.17}$$

With an initial basic feasible point $[x, w]^\top = [0,\ b]^\top$, we can solve eq. (3.17) by the simplex method and obtain an optimal solution $[x^*, w^*]$. If $w^* = 0$, then we go to phase II which is to solve the original problem with x^* as a feasible basic solution to start with. Otherwise, the original problem has no feasible solution [7, p. 39].

3.4 Bland's rule

When the simplex method solves degenerate problems, the method may repeat the same sequence of degenerate pivots associated with a non-optimal solution. In this case, the method is said to be *cycling* and will never terminate [7, §3.5].

To avoid cycling, we can apply *Bland's* rule to choose the entering and the leaving variables to a basis. The idea of Bland's rule is to determine the choice of both the entering and the leaving variables. To do so, we first order the variables $x_1, x_2, \ldots, x_n$. At each iteration, among all non-basic variables with negative reduced cost, the simplex method selects the entering variable with the smallest index to enter the basis. Next, Bland's rule breaks a tie in the ratio test by choosing the basic variable with the smallest index from all potential leaving variables to leave the basis [7, §3.6].

3.5 Summary

To conclude, we studied the simplex algorithm and the two-phase method to obtain initial basic feasible solutions. For an implementation, we used the revised simplex method as it needs less storage and less computation. We also discussed how the simplex method can avoid cycling when solving degenerate problems by applying Bland's rule.

For the next two chapters, we will look at two interior point methods: the affine scaling and the primal-dual methods. Each of them is currently considered to be an efficient general method for solving linear programs.

Chapter 4

Affine scaling methods

The affine scaling method was first introduced by Dikin in 1967 [10, p.336]. Unfortunately, his work was not widely well-known until several researchers independently rediscovered that the affine scaling method is a simple version of Karmarkar's algorithm [44].

In this chapter, we will describe the affine scaling method and its implementation in sections 4.1 and 4.2. Similar to the simplex method, the affine scaling needs an interior feasible solution to start with, therefore in section 4.3, we will explain how to find starting points.

4.1 Formulating affine scaling methods

The affine scaling method solves the linear program in the form of (P), that is,

$$\min c^\top x \text{ subject to } Ax = b \text{ and } x \geq 0. \tag{P}$$

The idea of the method is to apply the affine scaling transformation to keep iterative points away from the boundary so that it has a decent improvement. At each iteration, the method generates an interior feasible solution that decreases the

corresponding objective function value.

Let k be the number of iteration. Suppose that we have a strictly positive initial solution x^k which is not optimal, so we must find another solution denoted by

$$x^{k+1} = x^k + \beta d, \tag{4.1}$$

where $\beta > 0$ is a step length and d is a direction which decreases the objective function value. Note that a direction $d = -c$ is a good choice as it makes the objective function value decrease. To stay in the feasible region, the new point must satisfy

$$Ax^{k+1} = A(x^k + \beta d) = Ax^k + \beta Ad = b. \tag{4.2}$$

Therefore, the direction d must satisfy $Ad = 0$ which is equivalent to projecting d onto the null space of the matrix A [10, p.336]. Since A has full rank, the projection matrix

$$P_A = I - A^\top (AA^\top)^{-1} A \tag{4.3}$$

maps any direction d onto the null space of A [44, p.348]. Then, we project the proposed direction $-c$ onto the null space of A and obtain a direction:

$$d = -P_A c. \tag{4.4}$$

Note that $P_A^\top = P_A$ and $P_A^2 = P_A$, so

$$c^\top d = -c^\top P_A c = -c^\top P_A P_A c = -(P_A c)^\top (P_A c). \tag{4.5}$$

Therefore, $d = -P_A c$ also yields a decrease of the objective function value [44, p.348].

If $P_A c = 0$, then for any direction γ,

$$c^\top x^{k+1} = c^\top (x^k + \beta P_A \gamma) = c^\top x^k + \beta (P_A c)^\top \gamma = c^\top x^k. \tag{4.6}$$

In this case, the objective function value remains the same. Therefore, x^k is an optimal solution [7, Lemma 7.2, p.150].

Note that if a component of x^k is closer to the boundary, then the method can take only a small step without making x^k cross zero. Consequently, an improvement in the next iteration is very small. In fact, this issue can be avoided if x^k is near the centre of the feasible region. Therefore, we should scale x^k so that we can make a decently long step for the next iteration. We also scale the linear programming problem and compute the scaled direction.

We first formulate a scaled linear programming problem. The following part follows [10, §10.5]. Let $x^k > 0$ be an interior feasible point of (P). We scale each component by mapping each of them to unity so that each of them stays at least one unit away from zero. Let $X = \mathrm{diag}(x^k)$ be the $n \times n$ diagonal matrix whose diagonal elements are the components of x^k. Note that $X^\top = X$. As x^k is an interior point, the diagonal elements of matrix X are strictly positive and therefore, X is invertible. Since $Ax = AXX^{-1}x$ and $c^\top x = c^\top XX^{-1}x$, if we define

$$z = X^{-1}x, \tag{4.7}$$

then we have $z^k = X^{-1}x^k = e$ where e is a vector all of whose elements are equal one. We must scale A and c as:

$$\tilde{A} = AX \quad \text{and} \quad \tilde{c} = Xc. \tag{4.8}$$

Taking $z, \tilde{A}$ and $\tilde{c}$ back to (P), we obtain the scaled linear programming problem:

$$\min \quad \tilde{c}^{\top} z \quad \text{subject to} \quad \tilde{A}z = b \quad \text{and} \quad z \geq 0. \tag{$\tilde{P}$}$$

Note that x is feasible in the (P) problem if and only if z is feasible in the $(\tilde{P})$ problem and they have the same objective function value. With a step length $\beta > 0$, we move from a current scaled solution z^k to a new scaled solution:

$$z^{k+1} = z^k + \beta \frac{\tilde{d}}{||\tilde{d}||} \tag{4.9}$$

where $\tilde{d} = -P_{\tilde{A}}\tilde{c}$ is the direction of the next scaled iterate and $||v|| = (\sum_{i=1}^{n} v_i^2)^{\frac{1}{2}}$. Note that we divide $\tilde{d}$ by $||\tilde{d}||$ in eq. (4.9) to normalise the direction $\tilde{d}$ [33, p.118]. Again, if $P_{\tilde{A}}\tilde{c} = 0$, then by the same argument as in eq. (4.6), we achieve the optimal value. Finally, we transform the scaled solution z^{k+1} back to the original variable [33, p.118]:

$$x^{k+1} = X\left(z^k + \beta \frac{\tilde{d}}{||\tilde{d}||}\right) = x^k + \beta X \frac{\tilde{d}}{||\tilde{d}||}. \tag{4.10}$$

For any $\beta > 0$, we have

$$\begin{aligned}
c^{\intercal} x^{k+1} &= c^{\intercal} x^k + c^{\intercal}\left(\beta X \frac{\tilde{d}}{||\tilde{d}||}\right) && \text{(by eq. (4.10))} && (4.11)\\
&= c^{\intercal} x^k - \frac{\beta}{||\tilde{d}||}(\tilde{c}^{\intercal} P_{\tilde{A}} \tilde{c}) && \left(\text{by } \tilde{c} = Xc,\ \tilde{d} = -P_{\tilde{A}}\tilde{c}\right) && (4.12)\\
&= c^{\intercal} x^k - \frac{\beta}{||\tilde{d}||}(P_{\tilde{A}}\tilde{c})^{\intercal}(P_{\tilde{A}}\tilde{c}) && \left(\text{by } P_{\tilde{A}}^2 = P_{\tilde{A}}\right) && (4.13)\\
&= c^{\intercal} x^k - \frac{\beta}{||\tilde{d}||}\tilde{d}^{\intercal}\tilde{d} && \left(\text{by } \tilde{d} = -P_{\tilde{A}}\tilde{c}\right). && (4.14)
\end{aligned}$$

Therefore, we can detect unboundedness from the sign of $\tilde{d}$.

Lemma 14. [7, p.150] If $\tilde{d} \geq 0$ and $\tilde{d} \neq 0$, then the problem (P) is unbounded. □

Proof. Suppose $\tilde{d} \geq 0$ and $\tilde{d} \neq 0$, then for any $\beta > 0$, it is seen from eq. (4.14)

that $c^\intercal x^{k+1} = c^\intercal x^k - \frac{\beta}{||\tilde{d}||}\tilde{d}^\intercal\tilde{d} \to -\infty$ as $\beta \to \infty$. Therefore, the problem (P) is unbounded. □

On the other hand, if $\tilde{d}_j < 0$ for some j, then we choose step length β such that the new iterative solution is feasible. Indeed, we set $\beta := \alpha\beta^*$ where β^* is the largest step such that

$$\beta^* = \sup\{\beta : z^k + \beta\tilde{d} > 0\}, \tag{4.15}$$

and $0 < \alpha < 1$ is a step length factor. In this case, β^* is given by

$$\beta^* = \frac{||\tilde{d}||}{\max\limits_{\tilde{d}_j<0} |\tilde{d}_j|}. \tag{4.16}$$

[33, p.118]. The new scaled iterate z^{k+1} will be

$$z^{k+1} = z^k + \alpha\beta^*\frac{\tilde{d}}{||\tilde{d}||}. \tag{4.17}$$

Finally, using the relation $x = Xz$, we derive a new iterate

$$x^{k+1} = x^k + \alpha\beta^* X\frac{\tilde{d}}{||\tilde{d}||}. \tag{4.18}$$

4.2 Implementation

In this section, we look at step-size and stopping criteria for the affine scaling method.

Under various step-size α and non-degeneracy assumptions, several authors addressed convergence of a sequence of iterates generated by the algorithm (see [41] for a list of literature). With the non-degenerate assumptions, a sequence of iterates converges to the optimal solution without a restriction on the step size α [33, p.128]. In general, α is set very close to 1, for example, Griva et al. [10, p.338] suggest that

$\alpha = 0.99$.

Without the non-degeneracy assumption, Mascarenhas [21] showed a linear programming problem that has an optimal solution, but the sequence of iterates converges to a non-optimal solution when $\alpha = 0.999$. Terlaky and Tsuchiya [42] presented that for Mascarenhas' linear programming problem, the sequence of iterates of the affine scaling method does not converge to an optimal solution for any $\alpha \geq 0.91$. Tsuchiya and Muramatsu [41] ensured that with $\alpha \leq 2/3$, the sequence of iterates will converge to an optimal solution. Therefore, in our implementation to solve degenerate problems for checking avoiding sure loss, we choose $\alpha = 2/3$.

A stopping criterion is motivated by a decreasing of the objective function values. A commonly used criterion is the the relative improvement in the objective is small, that is, when

$$\frac{c^\top x^k - c^\top x^{k+1}}{\max\{1, |c^\top x^k|\}} < \epsilon \tag{4.19}$$

where ϵ is a positive tolerance value [10, p.338]. We can also add a lower bound to detect unboundedness. Let $M > 0$ be a very large number. If $c^\top x^k < -M$ for some k, then we say that the sequence $\{c^\top x^k\}$ is unbounded and x^k does not converge.

Summarising all these arguments, we outline the affine scaling method as follows:

Step 1 (initialisation): Find initial interior feasible point x^0. Set $k = 0$, $\alpha = 0.99$ ($\alpha = 2/3$ if the problem is degenerate), $\epsilon > 0$, and $M > 0$.

Step 2 (finding directions): Calculate a direction $\tilde{d} = -P_{\tilde{A}}\tilde{c}$. where $P_{\tilde{A}} = I - \tilde{A}^\top(\tilde{A}\tilde{A}^\top)^{-1}\tilde{A}$ is a projection matrix.

Step 3 (checking optimality via directions): If $\tilde{d} = 0$, then an optimal solution is x^k and the optimal value is $c^\top x^k$. If $\tilde{d} \geq 0$ and $\tilde{d} \neq 0$, then the problem is unbounded. In both cases, we stop here. Otherwise, we move to Step 4.

Step 4 (moving to new solutions): Calculate a step length $\beta^* = \dfrac{||\tilde{d}||}{\max\limits_{\tilde{d}_j<0} |\tilde{d}_j|}$ and a

new iteration $x^{k+1} = x^k + \alpha\beta^* X \frac{\tilde{d}}{||\tilde{d}||}$.

Step 5 (checking optimality via objective values): If $c^\top x^{k+1} < -M$, then the problem is unbounded. If

$$\frac{c^\top x^k - c^\top x^{k+1}}{\max\{1, |c^\top x^k|\}} < \epsilon,$$

then we find an optimal solution x^{k+1} and the optimal value $c^\top x^{k+1}$. In both cases, we stop here. Otherwise, we set $k = k + 1$ and return to Step 2.

4.3 Finding initial interior solutions

The affine scaling method requires an initial interior feasible solution to start with. Similar to the simplex method, we can find an initial interior feasible point by using the big-M or the two-phase methods [7, p. 155]. These techniques require solving an additional linear programming problem. Here, we explain how to apply the two-phase method to find an initial interior feasible point for the affine scaling method.

Consider constraints $Ax = b$ and $x \geq 0$. We choose any interior point $x^0 > 0$ and calculate $z \coloneqq b - Ax^0$. If $z = 0$, then x^0 is an interior feasible solution. Otherwise, we solve

$$\text{(phase-I)} \qquad \min \quad \gamma \tag{4.20}$$

$$\text{subject to} \quad Ax + z\gamma = b \tag{4.21}$$

$$\text{where} \quad x \geq 0,\ \gamma \geq 0 \tag{4.22}$$

by the affine scaling method with an interior feasible solution $[x\ \gamma] = [x^0\ 1]$. If we find an optimal solution $[x^*\ \gamma^*]$ such that $\gamma^* = 0$, then x^* is an interior feasible solution of the original problem. Otherwise, the original problem does not have an interior feasible solution [7, p.156].

We will use this technique to find interior feasible points later in section 7.4 where we solve linear programs for checking avoiding sure loss by the affine scaling method.

4.4 Summary

To conclude, we studied and discussed the affine scaling method which is one of the most commonly used interior point methods. As the method needs to start with an initial interior solution, we explained how to obtain starting points by applying the two-phase method. We also discussed a limitation on the step-size when the method solves degenerate problems.

In the next chapter, we will look at another interior-point method that does not have a limitation on the step-size and does not need to start with a feasible solution.

Chapter 5

Primal-dual methods

The primal-dual method is one of the most commonly used interior point methods for solving linear programs. The idea of the primal-dual method is based on applying a logarithmic barrier function. The primal-dual method can solve primal and dual problems simultaneously and the method is known to be a polynomial time algorithm [19, 25].

In this chapter, we first apply a logarithmic barrier function to formulate the primal-dual method in section 5.1 and then describe the primal-dual method in section 5.2. We summarise a practical implementation of the method in section 5.3.

5.1 Formulating primal-dual methods

Consider a pair of primal and dual linear programs:

$$\min c^\top x \text{ subject to } Ax = b,\ x \geq 0, \tag{P}$$

$$\max b^\top y \text{ subject to } A^\top y + t = c,\ t \geq 0,\ y \text{ free}. \tag{D}$$

Let x be a feasible solution to (P) and $[y\ t]$ be a feasible solution to (D). Remember that by theorem 13, these points are optimal if and only if the duality gap $x^\top t$ is

equal to zero. Since x and t are non-negative, $x^\top t$ is equal to zero if and only if for all $i = 1, \ldots, n$ either x_i or t_i is equal to zero.

The original idea of the primal-dual method is to optimise the (P) and (D) problems simultaneously by generating a sequence of interior feasible solutions $[x\ y\ t]$ (i.e. $x > 0$ and $t > 0$) that satisfies the complementary slackness condition. We assume that both (P) and (D) have feasible interior solutions.

For $x > 0$ in (P), we can apply the *logarithm barrier function* technique and then consider the following family of nonlinear programming problems:

$$(\mathrm{P}_\mu) \qquad \min \quad c^\intercal x - \mu \sum_{j=1}^{n} \log x_j \tag{5.1}$$

$$\text{subject to} \quad Ax = b,\ x > 0 \tag{5.2}$$

where $\mu > 0$ is a barrier parameter and this problem is indexed by μ. We will show that as $\mu \to 0$, the optimal solution of (P_μ) converges to an optimal solution of (P).

To see this, we first note that the objective function of (P_μ) is a strictly convex function, therefore (P_μ) has at most one global minimum. This can be obtained by solving the first-order optimality conditions (eqs. (5.6) and (5.7)). We apply the Lagrangian multiplier method to (P_μ). The Lagrangian function for (P_μ) is

$$L(x, y) = c^\intercal x - \mu \sum_{j=1}^{n} \log x_j - y^\intercal (Ax - b). \tag{5.3}$$

Taking derivative with respect to each variable and setting them equal to zero, we obtain the first-order optimality conditions:

$$\frac{\partial L}{\partial x_j} = c_j - \frac{\mu}{x_j} - \sum_{i=1}^{m} y_i a_{ij} = 0 \qquad j = 1, \ldots, n \tag{5.4}$$

$$\frac{\partial L}{\partial y_i} = b_i - \sum_{j=1}^{n} a_{ij} x_j = 0 \qquad i = 1, \ldots, m. \tag{5.5}$$

We rewrite these conditions in matrix form:

$$A^\intercal y + \mu X^{-1} e = c \tag{5.6}$$

$$Ax = b, \tag{5.7}$$

where $X = \text{diag}(x)$ denotes a diagonal matrix whose ith diagonal element is x_i and e denotes an $n-$vector whose elements are all one. Setting

$$t = \mu X^{-1} e, \tag{5.8}$$

eq. (5.6) can be written as

$$A^\intercal y + t = c. \tag{5.9}$$

Multiplying eq. (5.8) by X, we obtain the following system [7, p.178]:

$$(\text{PD}_\mu) \qquad Ax = b, \quad x > 0 \qquad \text{(primal feasiblity)} \tag{5.10}$$

$$A^\intercal y + t = c, \quad t > 0 \qquad \text{(dual feasiblity)} \tag{5.11}$$

$$XTe = \mu e, \qquad \text{(duality gap)}, \tag{5.12}$$

which determines the optimal solution of (P_μ). Note that eq. (5.12) can be rewritten as

$$x_j t_j = \mu \qquad j = 1, \ldots, n. \tag{5.13}$$

Therefore, given $\mu > 0$, x uniquely determines t by eq. (5.13) and y by eq. (5.11). Theorem 15 states that (PD_μ) has a unique solution:

Theorem 15. ([7, p.179]) Suppose that (P) and (D) have feasible solutions. Then both (PD_μ) and (P_μ) have a unique solution. □

A solution x of (P_μ) is exactly x in a solution $[x\ y\ t]$ of (PD_μ). If $\mu \to 0$, then the optimal solution of (PD_μ) converges to an optimal solution of (P) (also obtain an optimal solution of (D) as well). Therefore, we now consider solving (PD_μ).

For each $\mu > 0$, let $[x(\mu)\ y(\mu)\ t(\mu)]$ be the unique solution to (PD_μ). As $x(\mu)$ satisfies eq. (5.10), $x(\mu)$ is a feasible solution of (P). Similarly, $y(\mu)$ and $t(\mu)$ satisfy eq. (5.11), therefore $[y(\mu)\ t(\mu)]$ is a feasible solution of (D). In this case, the duality gap becomes

$$c^\top x(\mu) - b^\top y(\mu) = (c^\top - y(\mu)^\top A)x(\mu) \tag{5.14}$$

$$= t(\mu)^\top x(\mu) = n\mu, \tag{5.15}$$

where n is the length of x. As $\mu \to 0$, the duality gap also converges to zero. The following theorem states that the optimal solution of (PD_μ) converges to an optimal solution of (P):

Theorem 16. ([7, p.179]) Suppose that both (P) and (D) have feasible solutions. If $[x(\mu)\ y(\mu)\ t(\mu)]$ is the unique solution to (PD_μ), then as $\mu \to 0$,

1. $x(\mu)$ converges to an optimal solution of (P), and
2. $[y(\mu)\ t(\mu)]$ converges to an optimal solution of (D).

□

5.2 Primal-dual methods

The main computational task in the primal-dual method is to solve the system (PD_μ).

Let k be the number of iterations and let $[x^k\ y^k\ t^k]$ be a solution that satisfies the system (PD_μ^k) at some μ^k. If μ^k is not close to zero, then we must find a new solution for a smaller value of μ. To do so, we choose μ^{k+1} to be a smaller value by setting $\mu^{k+1} = \theta\mu^k$ for some $0 < \theta < 1$. Let the next iterative solutions be in the

form:

$$x^{k+1} = x^k + \beta_P d_x \tag{5.16}$$

$$y^{k+1} = y^k + \beta_D d_y \tag{5.17}$$

$$t^{k+1} = t^k + \beta_D d_t, \tag{5.18}$$

of the system $(PD_{\mu^{k+1}})$, where β_P and β_D are step lengths for primal and dual solutions, and d_x, d_y and d_t are directions. Our goal is to find these directions such that the new estimate points, $x^k + d_x,\ y^k + d_y,\ t^k + d_t$, stay in the feasible region. Note that we will come back to choose the step length later after we obtain the directions.

To start, we find the direction d_x that maintains the primal feasibility, that is,

$$Ax^{k+1} = A(x^k + d_x) = b, \tag{5.19}$$

and since $Ax^k = b$, we have $Ad_x = 0$. Similarly, the directions d_y and d_t satisfy the dual feasibility,

$$A^\top y^{k+1} + t^{k+1} = A^\top (y^k + d_y) + (t^k + d_t) = c. \tag{5.20}$$

Since $A^\top y^k + t^k = c$, we have $A^\top d_y + d_t = 0$. Again, substituting solutions x^k and t^k into eq. (5.12), we obtain

$$(x_i^k + d_x)(t_i^k + d_t) = \mu^{k+1} \tag{5.21}$$

which is

$$t_i^k d_x + x_i^k d_t + d_x d_t = \mu^{k+1} - x_i^k t_i^k. \tag{5.22}$$

If d_x and d_t are small, then the term $d_x d_t$ is much smaller than d_x and d_t. Therefore,

we can consider

$$t_i^k d_x + x_i^k d_t = \mu^{k+1} - x_i^k t_i^k. \tag{5.23}$$

Next, we calculate the directions d_x, d_y and d_t by solving the following linear system:

$$Ad_x = 0 \tag{5.24}$$

$$A^\top d_y + d_t = 0 \tag{5.25}$$

$$Td_x + Xd_t = \mu^{k+1}e - XTe. \tag{5.26}$$

Denote

$$D \coloneqq T^{-1}X \quad \text{and} \quad v(\mu^{k+1}) \coloneqq \mu^{k+1}e - XTe, \tag{5.27}$$

then, by tedious substitution, we obtain

$$d_y = -(ADA^\top)^{-1}AT^{-1}v(\mu^{k+1}) \tag{5.28}$$

$$d_t = -A^\top d_y \tag{5.29}$$

$$d_x = T^{-1}v(\mu^{k+1}) - Dd_t. \tag{5.30}$$

Similar to the affine scaling method, if $d_x > 0$ and $c^\top d_x < 0$, then the primal problem is unbounded. If $d_t > 0$ and $b^\top d_y > 0$, then the dual problem is unbounded [7, p.192].

5.3 Implementation

In order to save computational time, the values of μ should rapidly decrease. In practice, we can set

$$\mu^{k+1} = \theta\mu^k \tag{5.31}$$

for a fixed $0 < \theta < 1$. By eq. (5.12), we see that $\mu^{k+1}e - XTe < 0$. In our implementation, as suggested by Vanderbei [44, p.307], we set θ to be $1/10$.

However, if μ is dramatically decreasing, then x^{k+1} and t^{k+1} may no longer be positive. To guarantee the positivity of x^{k+1} and t^{k+1}, we choose β_P and β_D such that they are the largest steps for which $x^k + \beta_P d_x > 0$ and $t^k + \beta_D d_t > 0$. In this case, we explicitly choose

$$\beta_P = \alpha \cdot \left(\max_i \left\{ 1, -\frac{(d_x)_i}{x_i} \right\} \right)^{-1} \tag{5.32}$$

$$\beta_D = \alpha \cdot \left(\max_i \left\{ 1, -\frac{(d_t)_i}{t_i} \right\} \right)^{-1}, \tag{5.33}$$

for some $0 < \alpha < 1$. So far, this approach seems to work well in practice [7, p.190]. Note that there is no restriction on α as in the affine scaling method. We can set α to be close to 1, for example $\alpha = 0.99$.

An advantage of the primal-dual method is that, instead of starting with a feasible solution, the method can start with arbitrary points $x^k > 0$, y^k and $t^k > 0$.

Suppose that we have initial points $x^k > 0$, y^k and $t^k > 0$ which are not optimal. We can find the directions d_x, d_y and d_t that satisfy the system eqs. (5.19), (5.20) and (5.27) as before. In this case, eqs. (5.19) and (5.20) become

$$r_P \coloneqq Ad_x = b - Ax^k \tag{5.34}$$

$$r_D \coloneqq A^\intercal d_y + d_t = c - A^\intercal y^k - t^k, \tag{5.35}$$

where r_P and r_D are primal and dual residuals respectively. We perform the same analysis and obtain the following directions:

$$d_y = -(ADA^\intercal)^{-1}[AT^{-1}v(\mu^{k+1}) - ADr_D - r_P] \tag{5.36}$$

$$d_t = -A^\intercal d_y + r_D \tag{5.37}$$

$$d_x = T^{-1}v(\mu^{k+1}) - Dd_t. \tag{5.38}$$

Next, we find the step lengths by eqs. (5.32) and (5.33) and calculate new iterative

points as in eqs. (5.16), (5.17) and (5.18). Finally, the method will terminate when (i) the duality gap is less than a tolerance and (ii) both iterative solutions are feasible, that is, both r_P and r_D are close to zero [7, p.191].

We summarise an implementation of the primal-dual method:

Step 1 (initialization) Given an initial point (x^0, y^0, t^0), where $x^0 > 0$ and $t^0 > 0$. Let n be the length of x^0. Set $k = 0$, $\theta = 1/10$, $\alpha = 0.99$ and a small tolerance $\epsilon > 0$.

Step 2 (initial calculation) Compute

$$\mu^{k+1} = \frac{(x^k)^\top t^k}{n} \tag{5.39}$$

$$r_P = b - Ax^k \tag{5.40}$$

$$r_D = c - A^\top y^k - t^k. \tag{5.41}$$

Step 3 (checking for optimality) If

$$\mu^{k+1} < \epsilon, \quad ||r_P|| < \epsilon, \quad \text{and} \quad ||r_D|| < \epsilon, \tag{5.42}$$

then we stop here and x^k, y^k and t^k are optimal. Otherwise, we go to Step 4.

Step 4 (calculate directions) Compute

$$D = XT^{-1} \quad \text{and} \quad v(\mu^{k+1}) = \mu^{k+1}e - XTe \tag{5.43}$$

and the directions:

$$d_y = -(ADA^\top)^{-1}[AT^{-1}v(\mu^{k+1}) - ADr_D - r_P] \tag{5.44}$$

$$d_t = -A^\top d_y + r_D \tag{5.45}$$

$$d_x = T^{-1}(\mu^{k+1}) - Dd_t. \tag{5.46}$$

Step 5 (checking for unboundedness) If

$$||r_P|| < \epsilon, \quad d_x > 0 \ \text{ and } \ c^\top d_x < 0, \tag{5.47}$$

then the primal problem is unbounded, so we stop here. If

$$||r_D|| < \epsilon, \quad d_t > 0 \ \text{ and } \ b^\top d_y > 0, \tag{5.48}$$

then the dual problem is unbounded, therefore we stop here. Otherwise, we go to Step 6.

Step 6 (calculating step lengths) Compute the directions

$$\beta_P = \alpha \cdot \left(\max_i \left\{ 1, -\frac{(d_x)_i}{x_i} \right\} \right)^{-1} \tag{5.49}$$

$$\beta_D = \alpha \cdot \left(\max_i \left\{ 1, -\frac{(d_s)_i}{t_i} \right\} \right)^{-1}. \tag{5.50}$$

Step 7 (moving to new points) We update new points:

$$x^{k+1} = x^k + \beta_P d_x \tag{5.51}$$

$$y^{k+1} = y^k + \beta_D d_y \tag{5.52}$$

$$t^{k+1} = t^k + \beta_D d_t, \tag{5.53}$$

and go back to Step 2.

5.4 Summary

To summarise, the primal-dual method can simultaneously solve a pair of primal and dual problems. Unlike the simplex and the affine scaling methods, the primal-dual method does not need to start with feasible solutions and does not have any restriction on solving degenerate problems.

So far, we have seen three commonly used linear programming methods: the simplex, the affine scaling and the primal-dual methods. We will come back to these methods again in chapter 7 where we discuss how to improve these methods for efficiently solving linear programs for checking avoiding sure loss. Before that, we will explain the last basic concept: desirability axioms and lower previsions in chapter 6.

Chapter 6

Desirability and lower previsions

We begin this chapter by studying sets of desirable gambles and basic rationality criteria called rationality axioms for sets of desirable gambles in section 6.1 and avoiding sure loss in section 6.2. We also give examples in these two sections. Next, we discuss the natural extension in section 6.3 and in section 6.4, we discuss the Choquet integral which can be used to calculate natural extensions. We then study a special case of checking avoiding sure loss with adding one extra gamble into the sets that already avoids sure loss in section 6.5. We also discuss several types of lower previsions that are coherent in section 6.6. Finally, in section 6.7, we study problems of decision making with lower previsions.

6.1 Gambles and desirability axioms

Let Ω be a finite set of all possible outcomes of an experiment or an observation. These outcomes ω are assumed to be exhaustive and mutually exclusive. For example, in a football match between two teams, a result of the match for one team is either a win (W), a draw (D) or a loss (L). Therefore, a set of all possible outcomes is $\Omega = \{W, D, L\}$.

Assume that a subject would like to model her uncertainty about the outcomes of the match. If she has complete information about the match, then she may be able to model it through, e.g. in this case, a probability mass function. However, it may be difficult for her to specify precise probabilities because of having limited structural information about dependencies, lack of data, limited expert opinion, or even contradicting information from different experts. In that case, various authors [39, 46, 48, 49] have argued that these issues can be handled by modelling their beliefs using sets of desirable gambles.

A *gamble* is a bounded real-valued function on Ω. A gamble f represents an uncertain reward (e.g. money) that has value $f(\omega)$ to a subject if the true outcome is ω. A reward is expressed in units of utility. We denote the set of all gambles on Ω by $\mathcal{L}(\Omega)$.

Example 17. Considering a football match between Manchester United and Liverpool. Suppose that the subject is offered a reward depending on the outcomes of the match: she receives £10 if Manchester United wins (W), £5 for a draw (D) and nothing if Manchester United loses (D). These rewards can be viewed as a gambles f as follows:

Outcomes	W	D	L
f	10	5	0

□

From the above example, the subject should accept f since there is no loss in any outcome and she may gain money in some cases. Let $\mathcal{D}$ be a finite set of gambles that a subject is willing to accept; we call $\mathcal{D}$ the subject's *set of desirable gambles*. Suppose that a subject states her set of desirable gambles:

Outcomes	W	D	L
f_1	0	0	-1
f_2	10	0	-5
f_3	2	2	-5
f_4	12	2	-10

Even though the subject is pretty sure that the outcome L does not happen, f_1 should not be desirable to her because she will gain nothing and lose her money if the true outcome is L. Suppose that f_2 and f_3 are desirable to her. If someone offers her gamble f_4, then it should be desirable to her as well because accepting f_4 is similar to accept both f_2 and f_3.

By using utilities to represent the subject's reward, we satisfy scale invariance. For example, recall that the subject accepts f_2 in the previous table and suppose that she is offered another two gambles:

Outcomes	W	D	L
g_1	1000	0	-500
g_2	0.1	0	-0.05

Then, both g_1 and g_2, which are scaled from f_2, should also be desirable.

Taking all these arguments into account, we derive rationality conditions for desirability [39, p. 29]:

Axiom 1 (Rationality axioms for desirability)**.** For every f and g in $\mathcal{L}(\Omega)$ and every non-negative $\alpha \in \mathbb{R}$, we have that:

(D1) If $f \leq 0$ and $f \neq 0$, then f is not desirable.

(D2) If $f \geq 0$, then f is desirable.

(D3) If f is desirable, then so is αf.

(D4) If f and g are desirable, then so is $f + g$.

The first two axioms are trivial as the subject should accept any gamble that she cannot lose from, but she should not accept any gamble that she cannot gain from. Axiom (D3) follows the linearity of the utility scale and axiom (D4) shows that a combination of desirable gambles should also be desirable.

6.2 Avoiding sure loss

Even though the subject specifies a set of desirable gambles $\mathcal{D}$, it is not necessary that $\mathcal{D}$ satisfies all rationality axioms. However, we can use these axioms to examine the rationality of $\mathcal{D}$. Indeed, the rationality axioms essentially state that a non-negative combination of desirable gambles should not produce a sure loss [39, p. 30]. In that case, we say that $\mathcal{D}$ avoids sure loss.

Definition 18. [39, p. 32] A set $\mathcal{D} \subseteq \mathcal{L}(\Omega)$ is said to *avoid sure loss* if for all $n \in \mathbb{N}$, all $\lambda_1, \ldots, \lambda_n \geq 0$, and all $f_1, \ldots, f_n \in \mathcal{D}$,

$$\sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i f_i(\omega) \right) \geq 0. \tag{6.1}$$

□

We can also model uncertainty via acceptable buying (or selling) prices for gambles. Let $f \in \mathcal{L}(\Omega)$, the subject may be asked to specify how much she is willing to pay in order to get f. For instance, as in example 17, the subject is offered a gamble f. Suppose that she is willing to pay £5 to obtain f, then 5 can be viewed as a buying price for f. Therefore, her total reward will be:

Outcomes	W	D	L
$f - 5$	5	0	-5

It is obvious that she is willing to pay less than £5, but is she still willing to pay if it is more than £5? Given any $f \in \mathcal{L}(\Omega)$, we would like to know what is her highest

price for f? To answer this question, we have a function that represents a subject's highest buying price for f.

Definition 19. [39, p.30] A subject's *lower prevision* $\underline{P}$ is a real-valued function on $\operatorname{dom}\underline{P} \subset \mathcal{L}(\Omega)$, where $\operatorname{dom}\underline{P}$ is the domain of $\underline{P}$. □

Given a gamble $f \in \operatorname{dom}\underline{P}$, we interpret $\underline{P}(f)$ as a subject's supremum buying price for f, i.e. $f - \alpha$ is deemed desirable for all $\alpha < \underline{P}(f)$ [39, p. 40].

Any lower prevision $\underline{P}$ induces a *conjugate upper prevision* $\overline{P}$ on $\operatorname{dom}\overline{P} := \{-f\colon f \in \operatorname{dom}\underline{P}\}$, defined by $\overline{P}(f) := -\underline{P}(-f)$ for all $f \in \operatorname{dom}\overline{P}$. $\overline{P}(f)$ represents a subject's infimum selling price for f. This implies that the transaction $\alpha - f$ is desirable for all $\overline{P}(f) < \alpha$ [39, p. 41].

A lower prevision $\underline{P}$ is said to be *self-conjugate* when $\operatorname{dom}\underline{P} = -\operatorname{dom}\underline{P}$ and $\underline{P}(f) = \overline{P}(f)$ for all $f \in \operatorname{dom}\underline{P}$. We call a self-conjugate lower prevision $\underline{P}$ a *prevision* and write it as P [39, p. 41].

Given a lower prevision, we can construct a set of desirable gambles of $\underline{P}$ as follows [39, p.42]:

$$\mathcal{D}_{\underline{P}} := \{f - \mu : f \in \operatorname{dom}\underline{P} \text{ and } \mu < \underline{P}(f)\}. \tag{6.2}$$

Then, we can check whether $\underline{P}$ avoids sure loss or not through $\mathcal{D}_{\underline{P}}$:

Definition 20. [39, p.42] Let Ω be a finite set. We say that a lower prevision $\underline{P}$ avoids sure loss if one of the following equivalent conditions holds:

(i) The set of desirable gambles $\mathcal{D}_{\underline{P}}$ avoids sure loss.

(ii) For all $n \in \mathbb{N}$, all $\lambda_1, \ldots, \lambda_n \geq 0$, and all $f_1, \ldots, f_n \in \operatorname{dom}\underline{P}$,

$$\sup_{\omega\in\Omega}\left(\sum_{i=1}^{n}\lambda_i\left[f_i(\omega) - \underline{P}(f_i)\right]\right) \geq 0. \tag{6.3}$$

□

Proof. A proof of these two equivalent conditions is a part of the proof of six equivalent conditions in [39, p.42]. We prove directly that these two condition are equivalent.

($\Longrightarrow$) Suppose $\mathcal{D}_{\underline{P}}$ avoids sure loss. Let $n \in \mathbb{N}$, all $\lambda_1, \ldots, \lambda_n \geq 0$, and all $f_1, \ldots, f_n \in \operatorname{dom} \underline{P}$. We show that eq. (6.3) holds. Let $\epsilon > 0$. Then, for each i, $f_i - (\underline{P}(f_i) - \epsilon) \in \mathcal{D}_{\underline{P}}$. Since $\mathcal{D}_{\underline{P}}$ avoids sure loss,

$$\sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i \left[f_i(\omega) - \underline{P}(f_i) \right] \right) + \sum_{i=1}^{n} \lambda_i \epsilon \geq \sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i \left[f_i(\omega) - \underline{P}(f_i) + \epsilon \right] \right) \geq 0. \tag{6.4}$$

Since ϵ can be arbitrary small, eq. (6.3) holds.

($\Longleftarrow$) Suppose eq. (6.3) holds. We show that $\mathcal{D}_{\underline{P}}$ avoids sure loss. Let $\epsilon > 0$. Then, for $n \in \mathbb{N}$, all $\lambda_1, \ldots, \lambda_n \geq 0$ and for $i = 1, \ldots, n$, $f_i - (\underline{P}(f_i) - \epsilon) \in \mathcal{D}_{\underline{P}}$ and by eq. (6.3),

$$\sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i \left[f_i(\omega) - (\underline{P}(f_i) - \epsilon) \right] \right) \geq \sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i \left[f_i(\omega) - \underline{P}(f_i) \right] \right) \geq 0. \tag{6.5}$$

Therefore, $\mathcal{D}_{\underline{P}}$ avoids sure loss. □

For simplicity, we write $\mathcal{D}$ for $\mathcal{D}_{\underline{P}}$ when there is no confusion. Note that by definition 19, $\underline{P}$ avoids sure loss whenever $\mathcal{D}$ avoids sure loss.

Let us see some examples of avoiding sure loss.

Example 21. In the football match, suppose that the subject is offered gambles

Outcomes	W	D	L
f_1	20	10	5
f_2	10	15	5

If the subject specifies $\underline{P}(f_1) = 18$ and $\underline{P}(f_2) = 13$, then the total rewards will be:

Outcomes	W	D	L
$f_1 - \underline{P}(f_1)$	2	-8	-13
$f_2 - \underline{P}(f_2)$	-3	2	-8
$f_1 - \underline{P}(f_1) + f_2 - \underline{P}(f_2)$	-1	-6	-21

She is therefore certain to lose at least -1 regardless of the outcome which violates definition 20. Therefore, $\underline{P}$ does not avoid sure loss.

On the other hand, if the subject specifies $\underline{P}(f_1) = 5$ and $\underline{P}(f_2) = 5$, then her total rewards will be:

Outcomes	W	D	L
$f_1 - \underline{P}(f_1)$	15	5	0
$f_2 - \underline{P}(f_2)$	5	10	0

Since

$$\sup\{15\lambda_1 + 5\lambda_2, 5\lambda_1 + 10\lambda_2, 0\} \geq 0, \quad \forall \lambda_1, \lambda_2 \geq 0, \tag{6.6}$$

in this case, $\underline{P}$ avoids sure loss. □

Consider a special case of lower previsions. Let A denote a subset of Ω, also called an *event*. Its associated *indicator* function I_A is given by

$$\forall \omega \in \Omega\colon I_A(\omega) \coloneqq \begin{cases} 1 & \text{if } \omega \in A \\ 0 & \text{otherwise.} \end{cases} \tag{6.7}$$

In the chapter 10, we will extensively use *upper probability mass functions*. An upper probability mass function $\overline{p}$ is a mapping from Ω to $[0, 1]$, and represents the following lower prevision [39, p. 123]:

$$\forall \omega \in \Omega\colon \underline{P}_{\overline{p}}(-I_{\{\omega\}}) \coloneqq -\overline{p}(\omega), \tag{6.8}$$

where $\operatorname{dom} \underline{P}_{\overline{p}} = \bigcup_{\omega \in \Omega}\{-I_{\{\omega\}}\}$. We can check whether $\underline{P}_{\overline{p}}$ avoids sure loss by theorem 22.

Theorem 22. [39, p. 124] $\underline{P}_{\overline{p}}$ avoids sure loss if and only if $\sum_{\omega\in\Omega}\overline{p}(\omega) \geq 1$. □

Proof. See [39, p. 124, Prop. 7.2] with lower probability mass function $\underline{p} = 0$. □

We can interpret an upper probability mass function as providing an upper bound on the probability of each $\{\omega\}$, for all $\omega \in \Omega$ [39, p. 123].

6.3 Natural extension

The natural extension of a set of desirable gambles $\mathcal{D}$ is defined as the smallest set of gambles which includes all finite non-negative combinations of gambles in $\mathcal{D}$ and all non-negative gambles [39, § 3.7]:

Definition 23. [39, p. 32] The *natural extension* of a set $\mathcal{D} \subseteq \mathcal{L}(\Omega)$ is:

$$\mathcal{E}_{\mathcal{D}} \coloneqq \left\{ g_0 + \sum_{i=1}^{n} \lambda_i g_i \colon g_0 \geq 0,\, n \in \mathbb{N},\, g_1, \ldots, g_n \in \mathcal{D},\, \lambda_1, \ldots, \lambda_n \geq 0 \right\}. \tag{6.9}$$

□

From this natural extension $\mathcal{E}_{\mathcal{D}}$, we can derive a supremum buying price for any gamble $f \in \mathcal{L}(\Omega)$.

Definition 24. [39, p. 46] Let Ω be a finite set and let $\mathcal{D}$ be a set of desirable gambles. The natural extension $\underline{E}_{\mathcal{D}}$ defined on all $f \in \mathcal{L}(\Omega)$ is given by:

$$\begin{aligned} \underline{E}_{\mathcal{D}}(f) &\coloneqq \sup\{\alpha \in \mathbb{R} : f - \alpha \in \mathcal{E}_{\mathcal{D}}\} \\ &= \sup\left\{ \alpha \in \mathbb{R} : f - \alpha \geq \sum_{i=1}^{n} \lambda_i f_i, n \in \mathbb{N}, f_i \in \mathcal{D}, \lambda_i \geq 0 \right\}. \end{aligned} \tag{6.10}$$

□

Note that $\underline{E}_{\mathcal{D}}$ is finite, and hence, is a lower prevision, if and only if $\mathcal{D}$ avoids sure loss [39, p. 68]. We denote the conjugate of $\underline{E}_{\mathcal{D}}$ by $\overline{E}_{\mathcal{D}}$ which is defined on all

f in $\mathcal{L}(\Omega)$ by [46, p. 124]:

$$\overline{E}_{\mathcal{D}}(f) \coloneqq -\underline{E}_{\mathcal{D}}(-f) = \inf\left\{\beta \in \mathbb{R}\colon \beta - f \geq \sum_{i=1}^{n} \lambda_i f_i, n \in \mathbb{N}, f_i \in \mathcal{D}, \lambda_i \geq 0\right\}. \tag{6.11}$$

$\underline{E}_{\mathcal{D}}$ is simply denoted by $\underline{E}$ when there is no confusion.

Similarly, for any $f \in \mathcal{L}(\Omega)$, what does the lower prevision $\underline{P}$ defined on $\operatorname{dom}\underline{P}$ imply about the supremum buying price for f? Combining definition 24 and eq. (6.2) together, we can define the natural extension of $\underline{P}$:

Definition 25. [39, p.47] Let $\underline{P}$ be a lower prevision. The natural extension of $\underline{P}$ is defined for all $f \in \mathcal{L}(\Omega)$ by:

$$\begin{aligned}\underline{E}_{\underline{P}}(f) &\coloneqq \underline{E}_{\mathcal{D}_{\underline{P}}}(f) \\ &= \sup\left\{\alpha \in \mathbb{R} : f - \alpha \geq \sum_{i=1}^{n} \lambda_i (f_i - \underline{P}(f_i)), n \in \mathbb{N}, f_i \in \operatorname{dom}\underline{P}, \lambda_i \geq 0\right\}.\end{aligned} \tag{6.12}$$

□

Similarly, $\underline{E}_{\underline{P}}$ is finite if and only if $\underline{P}$ avoids sure loss [39, p. 68]. $\underline{E}_{\underline{P}}$ is simply denoted by $\underline{E}$ when there is no confusion.

6.4 Choquet integration

In this section, we briefly explain the Choquet integral which can be used to calculate the natural extension for the type of lower previsions considered later in chapter 10.

Let $\underline{E}_{\overline{p}}$ be the natural extension of $\underline{P}_{\overline{p}}$ that avoids sure loss. Because $\underline{E}_{\overline{p}}$ is 2-monotone, it can be computed via the Choquet integral [39, p. 125]. Based on the results from [39, Sec. 7.1], we give a closed form expression for this integral.

We simply denote the natural extension $\underline{E}_{\overline{p}}(I_A)$ of an indicator I_A as $\underline{E}_{\overline{p}}(A)$. We

can use the following theorem to calculate $\underline{E}_{\overline{p}}(A)$.

Theorem 26. [39, p. 125] Let $\underline{P}_{\overline{p}}$ avoid sure loss. Then for all $A \subseteq \Omega$,

$$\underline{E}_{\overline{p}}(A) = \max\{0, 1 - U(A^{\mathsf{c}})\} \qquad \text{and} \qquad \overline{E}_{\overline{p}}(A) = \min\{U(A), 1\}, \tag{6.13}$$

where $U(A) \coloneqq \sum_{\omega \in A} \overline{p}(\omega)$. □

Proof. See [39, p. 125] with lower probability mass function $\underline{p} = 0$. □

Theorem 27. Let f be decomposed in terms of its level sets A_i, $i = 0, 1, \ldots, n$:

$$f = \sum_{i=0}^{n} \lambda_i I_{A_i} \tag{6.14}$$

where $\lambda_0 \in \mathbb{R}$, $\lambda_1, \ldots, \lambda_n > 0$ and $\Omega = A_0 \supset A_1 \supset \cdots \supset A_n \neq \emptyset$. Then

$$\underline{E}_{\overline{p}}(f) = \sum_{i=0}^{n} \lambda_i \underline{E}_{\overline{p}}(A_i). \tag{6.15}$$

□

Proof. The right hand side is the Choquet integral [39, p. 379, Eq. (C.8)] and the natural extension $\underline{E}_{\overline{p}}(f)$ is equal to the Choquet integral [39, p. 125, Prop. 7.3(ii)] (with lower probability mass function $\underline{p} = 0$). □

Note that theorem 27 also holds for the upper natural extension.

Corollary 28. Let f be a gamble decomposed as in eq. (6.14). Then

$$\overline{E}_{\overline{p}}(f) = \sum_{i=0}^{n} \lambda_i \overline{E}_{\overline{p}}(A_i). \tag{6.16}$$

□

Proof. See appendix B.1. □

6.5 Avoiding sure loss with one extra gamble

Let $\mathcal{D} = \{g_1, \ldots, g_n\}$ be a finite set of desirable gambles that avoids sure loss and let f be another desirable gamble. We want to check whether $\mathcal{D} \cup \{f\}$ still avoids sure loss or not. This concept will be used to check avoiding sure loss of betting odds with a free coupon in section 10.2.

By the condition of avoiding sure loss in definition 18, $\mathcal{D} \cup \{f\}$ avoids sure loss if and only if for all $\lambda_0 \geq 0$, $n \in \mathbb{N}$, $g_i \in \mathcal{D}$ and $\lambda_1, \ldots, \lambda_n \geq 0$,

$$\max_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i g_i(\omega) + \lambda_0 f(\omega) \right) \geq 0. \tag{6.17}$$

We can simplify eq. (6.17) as follows.

Lemma 29. [29, Lemma 1] Let Ω be a finite set. Suppose that $\mathcal{D} = \{g_1, \ldots, g_n\}$ is a set of desirable gambles that avoids sure loss and f is another desirable gamble. Then, $\mathcal{D} \cup \{f\}$ avoids sure loss if and only if for all $n \in \mathbb{N}$, $g_i \in \mathcal{D}$ and $\lambda_1, \ldots, \lambda_n \geq 0$,

$$\max_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i g_i(\omega) + f(\omega) \right) \geq 0. \tag{6.18}$$

□

Proof. If $\lambda_0 = 0$, then eq. (6.17) is trivially satisfied as $\mathcal{D}$ avoids sure loss. Otherwise $\lambda_0 > 0$, and for all i, $\lambda_i \geq 0$, so $\lambda_i / \lambda_0 \geq 0$. Therefore eq. (6.17) is equivalent to

$$\max_{\omega \in \Omega} \left(\sum_{i=1}^{n} \left(\frac{\lambda_i}{\lambda_0} \right) g_i(\omega) + f(\omega) \right) \geq 0. \tag{6.19}$$

Therefore, $\mathcal{D} \cup \{f\}$ avoids sure loss if and only if eq. (6.18) holds. □

Next, we give a method not only for checking avoiding sure loss of $\mathcal{D} \cup \{f\}$, but also for bounding the worst case loss, which will be useful later in section 10.2.

Theorem 30. [29, Theorem 4] Let $f \in \mathcal{L}(\Omega)$ and let $\mathcal{D} = \{g_1, \ldots, g_n\}$ be a set of

desirable gambles that avoids sure loss. Then, $\mathcal{D} \cup \{f\}$ avoids sure loss if and only if $\overline{E}_{\mathcal{D}}(f) \geq 0$. If $\mathcal{D} \cup \{f\}$ does not avoid sure loss, then there exist $\lambda_1 \geq 0, \ldots, \lambda_n \geq 0$ such that $f + \sum_{i=1}^{n} \lambda_i g_i$, which is a combination of desirable gambles, results in a loss at least $|\overline{E}_{\mathcal{D}}(f)|$. □

Proof. See appendix B.2. □

Note that by definition 25, theorem 30 can also be applied to $\overline{E}_{\underline{P}}$.

6.6 Coherence

Coherence is another rationality condition for lower previsions and is stronger than avoiding sure loss. Coherence requires that the subject's supremum buying prices for gambles cannot be increased by considering any finite non-negative linear combination of other desirable gambles [46, §2.5.2]. In chapter 8, we will use coherent lower previsions to generate sets of desirable gambles that avoid sure loss.

Definition 31. [46, §2.5.4] A lower prevision $\underline{P}$ is said to be *coherent* if for all $n \in \mathbb{N}$, all $\lambda_0, \ldots, \lambda_n \geq 0$ and all $f_0, \ldots, f_n \in \operatorname{dom} \underline{P}$,

$$\sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i [f_i(\omega) - \underline{P}(f_i)] - \lambda_0 [f_0(\omega) - \underline{P}(f_0)] \right) \geq 0. \tag{6.20}$$

□

Next, we give some examples of coherent lower previsions. The lower prevision given by $\underline{P}(f) \coloneqq \inf f$ for all $f \in \mathcal{L}(\Omega)$ is coherent, and is called the *vacuous* lower prevision [46, §2.3.7]. Previsions that avoid sure loss are also coherent:

Theorem 32. [46, p.87] A prevision P is coherent if and only if it avoids sure loss (as a lower prevision). □

Therefore, for previsions, there is no difference between coherence and avoiding sure loss. The expectation of f associated with the probability mass function p is given by

$$E_p(f) \coloneqq \sum_{\omega \in \Omega} p(\omega) f(\omega). \tag{6.21}$$

An expectation operator E_p is coherent as well.

Generating probability mass functions is easy (see algorithm 1 further in section 8.1), and we can use them to generate other coherent lower previsions via lower envelopes and convex combinations:

Definition 33. [39, p. 60] Let Γ be a non-empty collection of lower previsions defined on a common domain $\mathcal{K}$. A lower prevision $\underline{Q}$ is called the *lower envelope* of Γ if

$$\underline{Q}(f) = \inf_{\underline{P} \in \Gamma} \underline{P}(f) \text{ for all } f \in \mathcal{K}. \tag{6.22}$$

□

Theorem 34. [39, p. 61] If all lower previsions in Γ are coherent, then the lower envelope of Γ is also coherent. □

We define the unit simplex as the set of all probability mass functions:

$$\Delta(\Omega) \coloneqq \left\{ p \in \mathbb{R}^{\Omega} \colon p \geq 0 \text{ and } \sum_{\omega \in \Omega} p(\omega) = 1 \right\}. \tag{6.23}$$

Its extreme points are the $\{0,1\}$-valued probability mass functions [46, §3.2.6]. The *credal set* of a lower prevision $\underline{P}$ is defined by

$$\mathcal{M}_{\underline{P}} := \{p \in \Delta(\Omega) : \forall f \in \operatorname{dom} \underline{P},\ E_p(f) \geq \underline{P}(f)\}. \tag{6.24}$$

$\mathcal{M}_{\underline{P}}$ completely determines $\underline{P}$ if $\underline{P}$ is coherent and there is a one-to-one correspondence between coherent lower previsions on $\mathcal{L}(\Omega)$ and closed convex subsets of $\Delta(\Omega)$ [39, p. 79]. Moreover, it suffices to consider the set of extreme points $\operatorname{ext} \mathcal{M}_{\underline{P}}$ of $\mathcal{M}_{\underline{P}}$ [46, p. 145]:

Theorem 35. (Adapted from [46, p. 146]) Let $\underline{P}$ be a coherent lower prevision. Then for every $f \in \operatorname{dom} \underline{P}$, there is a $p \in \operatorname{ext} \mathcal{M}_{\underline{P}}$ such that $\underline{P}(f) = E_p(f)$. □

When $\operatorname{ext} \mathcal{M}_{\underline{P}}$ is finite, then $\underline{P}$ is called *polyhedral*. We can construct a polyhedral lower prevision as follows. Let M be a finite set of probability mass functions on Ω. A polyhedral lower prevision is then given by [2, §9.2.1]

$$\underline{E}_M(f) \coloneqq \min_{p \in M} E_p(f). \tag{6.25}$$

Theorem 36. [46, p. 79] Let $\underline{P}_1$ and $\underline{P}_2$ be lower previsions on the same domain and $\delta \in [0, 1]$. If $\underline{P}_1$ and $\underline{P}_2$ are coherent, then so is $(1 - \delta)\underline{P}_1 + \delta \underline{P}_2$. □

Let P_0 be a coherent prevision on $\mathcal{L}(\Omega)$ and $\delta \in [0, 1]$. The lower prevision defined on all $f \in \mathcal{L}(\Omega)$ by

$$\underline{P}(f) \coloneqq (1 - \delta)P_0(f) + \delta \inf f \tag{6.26}$$

is called a *linear-vacuous mixture* [46, §2.9.2], and is coherent by theorem 36.

Note that we will use these coherent lower previsions to generate sets of desirable gambles later in chapter 8.

6.7 Decision making with lower previsions

In this section, we study three decision criteria: maximality, E-admissibility, and interval dominance. To see more about the relation between these criteria and their advantages and disadvantages, we refer to Huntley et al. [12] and Troffaes [38].

We first define two strict partial orders on $\mathcal{L}(\Omega)$, and then define optimality through maximality with respect to either of these two strict partial orderings.

Definition 37. For any two gambles f and g, we say that $f \succ g$ whenever

$$\underline{E}(f - g) > 0. \tag{6.27}$$

□

Note that Walley [46, §3.8.1] uses a stronger ordering, which also includes pointwise dominance. For this study, we follow Troffaes [38], Troffaes and Hable [40, §16.3.2] and Jansen et al. [13] and simply omit pointwise dominance from our definition.

Definition 38. [12, p. 194] For any two gambles f and g, we say that $f \sqsupset g$ whenever

$$\underline{E}(f) > \overline{E}(g). \tag{6.28}$$

□

Given any strict partial order on $\mathcal{L}(\Omega)$, we can define a notion of optimality through maximality with respect to that order:

Definition 39. Let $>$ be a strict partial order on $\mathcal{L}$, and let $\mathcal{K}$ be a finite subset of $\mathcal{L}(\Omega)$. The *set of maximal gambles in* $\mathcal{K}$ *with respect to* $>$ is then defined by:

$$\mathrm{opt}_{>}(\mathcal{K}) \coloneqq \{f \in \mathcal{K} \colon (\forall g \in \mathcal{K})(g \not> f)\}, \tag{6.29}$$

□

We call $\mathrm{opt}_{\succ}(\mathcal{K})$ the set of *maximal* gambles in $\mathcal{K}$ and $\mathrm{opt}_{\sqsupset}(\mathcal{K})$ the set of *interval dominant* gambles in $\mathcal{K}$.

Finally, we also need to define E-admissibility, which is yet another decision criterion. Recall that $\mathcal{M}_{\underline{P}}$ is the credal set of a lower prevision $\underline{P}$.

Definition 40 (E-admissibility)**.** [40, p. 336] A gamble f is *E-admissible* in $\mathcal{K}$ if

there is $p \in \mathcal{M}_{\underline{P}}$ such that

$$\forall g \in \mathcal{K} \colon E_p(f) \geq E_p(g). \tag{6.30}$$

□

The set of all E-admissible gambles in $\mathcal{K}$ is denoted by $\text{opt}_{\mathcal{M}}(\mathcal{K})$.

Note that [38]:

$$\text{opt}_{\mathcal{M}}(\mathcal{K}) \subseteq \text{opt}_{\succ}(\mathcal{K}) \subseteq \text{opt}_{\sqsupset}(\mathcal{K}). \tag{6.31}$$

If f is an E-admissible gamble in $\mathcal{K}$, then f is immediately maximal [46, §3.9.4]. If a gamble is not interval dominant, then it is not maximal. Consequently, if there are many gambles in the set, one may want to eliminate non-maximal gambles in $\mathcal{K}$ by applying interval dominance first [38].

6.8 Summary

To summarise, in this chapter, we briefly reviewed gambles, rationality axioms for desirability and avoiding sure loss. We also discussed lower previsions, natural extensions and the Choquet integral that can be used to calculate natural extensions. We studied coherence which will be used later in chapter 8 to generate sets of gambles that avoid sure loss. Finally, we studied several decision criteria with lower previsions. These basic concepts will be useful for the next three contributions in the thesis.

Part II

Contributions

Contribution I

Chapter 7

Improving algorithms for checking avoiding sure loss

This chapter is based on [26, §3–6] and [27, §3]. In this chapter, we first discuss several linear programs for checking avoiding sure loss. Based on the degenerate structure of the linear program, we slightly reduce the dimension and propose an extra stopping criterion. By looking at the three methods mentioned in chapters 3, 4 and 5, we analyse how we can solve linear programs for checking avoiding sure loss most effectively. To do so, we exploit the structure of the problems and also the interactions between the structure and the details of the algorithms. We present a direct way to obtain feasible starting points in various cases. This will reduce the effort required in the pre-solved phase of some of these algorithms.

7.1 Linear programming problems for checking avoiding sure loss

We study linear programming problems for checking avoiding sure loss. The problems (P1) and (D1) in theorem 41 are similar to the linear programming problems

discussed for lower previsions in [46, p. 175], which was further studied and extended by various authors [47, p.133].

Theorem 41. The set $\mathcal{D} = \{f_1, \ldots, f_n\}$ avoids sure loss if and only if the optimal value of (P1) is zero:

$$\begin{aligned}
\text{(P1)} \qquad & \min && \alpha && \text{(P1a)}\\
& \text{subject to} && \forall \omega \in \Omega\colon \sum_{i=1}^{n} f_i(\omega)\lambda_i - \alpha \leq 0 && \text{(P1b)}\\
& && \forall i\colon \lambda_i \geq 0 \quad (\alpha \text{ free}), && \text{(P1c)}
\end{aligned}$$

or, equivalently, if and only if its dual problem, (D1), has a feasible solution:

$$\begin{aligned}
\text{(D1)} \qquad & \max && 0 && \text{(D1a)}\\
& \text{subject to} && \forall f_i \in \mathcal{D}\colon \sum_{\omega \in \Omega} f_i(\omega) p(\omega) \geq 0 && \text{(D1b)}\\
& && \sum_{\omega \in \Omega} p(\omega) = 1 && \text{(D1c)}\\
& && \forall \omega\colon p(\omega) \geq 0. && \text{(D1d)}
\end{aligned}$$

□

Note that (P1) is fully degenerate, and (D1) is nearly fully degenerate. In fact, $\max 0$ in eq. (D1a) does not mean that anything is actually to be maximised but this statement is included to show how this dual problem fits into the standard pair of linear programs. The real question here is whether a feasible solution exists for (D1). Clearly, any feasible solution of (D1) is also an optimal solution, since the objective function is constant. A similar convention applies to (D2), (D4) and (D5).

7.2 Reduced sizes

As the simplex and the affine scaling methods require all variables in linear programming problems to be non-negative, we present alternative linear programming problems which are slightly smaller in dimension and have only non-negative variables. We presented the following theorem in [27, Theorem 6]:

Theorem 42. Choose any $\omega_0 \in \Omega$. The set $\mathcal{D} = \{f_1, \ldots, f_n\}$ avoids sure loss if and only if the optimal value of (P2) is zero:

$$\begin{aligned}
\text{(P2)} \qquad & \min && \sum_{i=1}^{n} \lambda_i f_i(\omega_0) + \alpha && \text{(P2a)}\\
& \text{subject to} && \forall \omega \neq \omega_0 \colon \sum_{i=1}^{n} (f_i(\omega_0) - f_i(\omega))\lambda_i + \alpha \geq 0 && \text{(P2b)}\\
& && \forall i \colon \lambda_i \geq 0 \text{ and } \alpha \geq 0, && \text{(P2c)}
\end{aligned}$$

or, equivalently, if and only if its dual problem, (D2), has a feasible solution:

$$\begin{aligned}
\text{(D2)} \qquad & \max && 0 && \text{(D2a)}\\
& \text{subject to} && \forall f_i \in \mathcal{D} \colon \sum_{\omega \neq \omega_0} (f_i(\omega_0) - f_i(\omega))p(\omega) \leq f_i(\omega_0) && \text{(D2b)}\\
& && \sum_{\omega \neq \omega_0} p(\omega) \leq 1 && \text{(D2c)}\\
& && \forall \omega \colon p(\omega) \geq 0. && \text{(D2d)}
\end{aligned}$$

□

Proof. See appendix B.3. □

Note that (P2) is still fully degenerate, whilst (D2) is no longer degenerate unless $f_i(\omega_0) = 0$ for some ω_0.

As primal optimality corresponds to dual feasibility [9, p. 104], if we choose any ω_0 such that most values $f_i(\omega_0)$ are non-negative, then (D2) will be closer to a dual

feasible solution, and therefore (P2) will also be closer to a primal optimal solution. For instance, if there is an ω_0 for which $f_i(\omega_0) \geq 0$ for all i, then we directly obtain a feasible solution of (D2) by setting $p(\omega) = 0$ for all $\omega \neq \omega_0$ [26]. The corresponding optimal solution of (P2) is given by $\lambda_i = 0$ for all i and $\alpha = 0$.

Next, we discuss how we can improve each linear programming algorithm for checking avoiding sure loss. Specifically, for each method, we pose problems (P2) and (D2) in a suitable format of linear programs, and we discuss an improvement for checking avoiding sure loss.

7.3 Improving simplex algorithms for checking avoiding sure loss

The whole investigation in this section closely follows [26, §5] and [27, §4.1]. Recall that to solve linear programs by the simplex method, we need a basic feasible solution. We can easily obtain a basic feasible solution of (P2) by multiplying eq. (P2b) by -1 and then adding a non-negative slack variable $s(\omega)$:

$$\text{(P3)} \qquad \min \quad \sum_{i=1}^{n} \lambda_i f_i(\omega_0) + \alpha \tag{P3a}$$

$$\text{subject to} \quad \forall \omega \neq \omega_0 : \sum_{i=1}^{n} (f_i(\omega) - f_i(\omega_0))\lambda_i - \alpha + s(\omega) = 0 \tag{P3b}$$

$$\forall i : \lambda_i \geq 0, \ \forall \omega \neq \omega_0 : s(\omega) \geq 0 \text{ and } \alpha \geq 0. \tag{P3c}$$

In this case, an initial basic feasible solution is given by setting all λ_i, α and $s(\omega) = 0$. Since all the right hand side of the constraints are zero, there is only one extreme point and that extreme point is zero. Also note that (P3) is fully degenerate, so the minimum ratios are all zero. Therefore, we apply Bland's rule to break a tie in the ratio test.

We now consider the dual problem (D2) and find a basic feasible starting point. To start, we add non-negative variables s_j to eqs. (D2b) and (D2c) and obtain equality constraints. To make all the right hand side values become non-negative, for every j such that $f_j(\omega) < 0$, we multiply the corresponding constraint by -1 to make it become non-negative and then add a slack variable $v_j \geq 0$. In this case, the size of the linear programming problem (D3) is slightly bigger:

$$\begin{aligned}
\text{(D3)} \qquad \min \quad & \sum_{j \in N} v_j && \text{(D3a)} \\
\text{subject to} \quad & \forall j \in N : \sum_{\omega \neq \omega_0} (f_j(\omega) - f_j(\omega_0)) p(\omega) - s_j + v_j = -f_j(\omega_0) && \text{(D3b)} \\
& \forall j \in I \setminus N : \sum_{\omega \neq \omega_0} (f_j(\omega_0) - f_j(\omega)) p(\omega) + s_j = f_j(\omega_0) && \text{(D3c)} \\
& \sum_{\omega \neq \omega_0} p(\omega) + q = 1 && \text{(D3d)} \\
& \forall \omega \neq \omega_0 : p(\omega) \geq 0,\ \forall j : s_j \geq 0,\ v_j \geq 0 \text{ and } q \geq 0 && \text{(D3e)}
\end{aligned}$$

with $I \coloneqq \{1, \ldots, n\}$ and $N \coloneqq \{j \in N \colon f_j(\omega_0) < 0\}$.

An initial extreme point for (D3) is given by $v_j = -f_j(\omega_0), s_j = 0$ for all $j \in N$, $s_j = f_j(\omega_0)$ for all $j \in I \setminus N$, $p(\omega) = 0$ for all $\omega \neq \omega_0$ and $q = 1$. If all $f_j(\omega_0)$ are non-negative, then we immediately solve the problem. Indeed, (D3) is normally non-degenerate unless $f_j(\omega_0) = 0$, for some j.

To sum up, in order to check avoiding sure loss by the simplex method, we can solve either (P3), which is fully degenerate, or (D3), whose size is slightly larger. Although the simplex method may stall under degeneracy, in practice, the method is one of the most commonly used algorithms for solving linear programs. Therefore, we consider the simplex method for checking avoiding sure loss as well.

7.4 Improving affine scaling methods for checking avoiding sure loss

We now look at the affine scaling method for checking avoiding sure loss. Again, we first pose both problems (P2) and (D2) in the suitable format of linear programs for the affine scaling method. Next, we discuss an impact of degeneracy to the performance of the affine scaling method. As the method requires initial interior feasible points, we also give direct ways to obtain such points. Note that the whole investigation in this section closely follows [26, §6] and [27, §4.2].

Recall that the affine scaling method solves the linear programming problem in the form of (P), that is, $\min c^\top x$ subject to $Ax = b$ and $x \geq 0$. For (P2), we simply pose it the form of (P3). For the dual problem (D2), it can be written in the form of (P) by adding non-negative slack variables:

$$\begin{aligned}
\text{(D4)} \qquad \max \quad & 0 && \text{(D4a)}\\
\text{subject to} \quad & \forall f_i \in \mathcal{D}\colon \sum_{\omega \neq \omega_0} (f_i(\omega_0) - f_i(\omega))p(\omega) + t_i = f_i(\omega_0) && \text{(D4b)}\\
& \sum_{\omega \neq \omega_0} p(\omega) + q = 1 && \text{(D4c)}\\
& \forall \omega \neq \omega_0\colon p(\omega) \geq 0,\ \forall i\colon t_i \geq 0,\ q \geq 0. && \text{(D4d)}
\end{aligned}$$

Even though we can also solve (D3) by the affine scaling method, we prefer to solve (D4) as it has fewer artificial variables.

As the problem (P3) is fully degenerate, similar to the simplex method, degeneracy can affect the performance of the affine scaling method. Specifically, the step-size is not larger than 2/3 [41]. However, unlike the simplex method, the affine scaling method can apply lemma 9. Indeed, when the affine scaling method solves (P3), the method can stop as soon as it finds a negative objective function value [27].

Remember that we can find an initial interior feasible point by using the two-phase method. Therefore, in general, the affine scaling normally solves two linear programming problems: one to find a starting interior feasible point, and another one to solve the original problem with this starting point.

Fortunately, to each one of (P3) and (D4), the affine scaling method needs to solve only one linear program. Specifically, for (D4), we only check whether the problem has a feasible solution or not. This is because every interior feasible point is also an optimal solution, so we can apply the two-phase method and only need to solve eq. (3.17). For (P3), due to the structure of the problem, we can immediately write down an interior feasible point in closed form, therefore in this case, we do not have to solve eq. (3.17).

Allow us explain how eq. (3.17) looks in the case of (D4). Let $\Omega \setminus \{\omega_0\} = \{\omega_1, \ldots, \omega_m\}$. Consider, for the moment, an arbitrary

$$x^0 = \begin{bmatrix} p^0(\omega_1) & \cdots & p^0(\omega_m) & t_1^0 & \cdots & t_n^0 & q^0 \end{bmatrix} > 0 \tag{7.1}$$

and define $[r \; z] \coloneqq b - Ax^0$, so

$$r_i \coloneqq h_i - t_i^0 \tag{7.2}$$

$$z \coloneqq 1 - \left(\sum_{\omega \neq \omega_0} p^0(\omega) + q^0 \right) \tag{7.3}$$

where

$$h_i \coloneqq f_i(\omega_0) - \sum_{\omega \neq \omega_0} (f_i(\omega_0) - f_i(\omega)) p^0(\omega). \tag{7.4}$$

If we choose $q^0 = p^0(\omega) = 1/|\Omega|$ for all $\omega \neq \omega_0$, then $z = 0$. Choose $t_i^0 = 1$ (or any other strictly positive value) for all i where $h_i \leq 0$. Finally, choose $t_i^0 = h_i$ for all i

where $h_i > 0$, so all corresponding r_i are zero. So, eq. (3.17) for (D4) becomes [27]:

$$\begin{aligned}
\text{(D4')} \qquad \min \quad & \gamma && \text{(D4'a)}\\
\text{subject to} \quad & \forall i\colon \sum_{\omega\neq\omega_0} (f_i(\omega_0) - f_i(\omega))p(\omega) + t_i + r_i\gamma = f_i(\omega_0) && \text{(D4'b)}\\
& \sum_{\omega\neq\omega_0} p(\omega) + q = 1 && \text{(D4'c)}\\
& \forall \omega \neq \omega_0\colon p(\omega) \geq 0,\ \forall i\colon t_i \geq 0,\ q \geq 0 \text{ and } \gamma \geq 0, && \text{(D4'd)}
\end{aligned}$$

with an initial interior feasible point as constructed. Note that for simplicity in our implementation later, we choose $t_i^0 = 1$ for all i. If the optimal solution of (D4') has $\gamma^* = 0$, then we will have found an interior feasible solution for (D4) (and therefore also an optimal solution for (D4)), and so $\mathcal{D}$ avoids sure loss; otherwise, there is no feasible solution and $\mathcal{D}$ incurs sure loss [27].

For (P3), we simply obtain a starting interior feasible point using theorem 43 below, with $\lambda_i^0 = 1$ for all i.

Theorem 43. [27, Theorem 7] An interior feasible solution of the following system of linear constraints

$$\forall j \in \{1, \ldots, m\}\colon \sum_{i=1}^{n} a_{ij}\lambda_i - \alpha + s_j = b_j \tag{7.5}$$

$$\forall i\colon \lambda_i \geq 0,\ \forall j\colon s_j \geq 0,\ \alpha \geq 0 \tag{7.6}$$

is given by setting $\lambda_i = \lambda_i^0$ for some arbitrary $\lambda_i^0 > 0$, $\alpha = 1 + \max\{0, -\delta\}$ with

$$\delta \coloneqq \min_j \left\{ b_j - \sum_{i=1}^{n} a_{ij}\lambda_i^0 \right\}, \tag{7.7}$$

and $s_j = b_j - \sum_{i=1}^{n} a_{ij}\lambda_i^0 + \alpha$. □

Proof. We must show that eq. (7.5) is satisfied, and that all variables are strictly positive.

Clearly, eq. (7.5) is satisfied by our choice of s_j, all $\lambda_i = \lambda_i^0 > 0$, and $\alpha \geq 1 > 0$. Finally, all $s_j > 0$ because

$$s_j = b_j - \sum_{i=1}^{n} a_{ij}\lambda_i^0 + \alpha \geq \delta + \alpha \geq \delta + 1 - \delta > 0, \tag{7.8}$$

where we used the definitions of δ and α respectively. □

To summarise, to check avoiding sure loss with the affine scaling method, we either solve (P3) or (D4'). In either case, we have a closed form for obtaining an initial interior feasible point. In addition, when solving (P3), we can apply lemma 9 as an extra stopping criterion to detect unboundedness. However, we have a limitation on the step-size due to degeneracy.

Next, we look at the primal-dual method for which we can also apply lemma 9 and theorem 43, but which does not have a limitation on the step-size and which has faster convergence as observed in practice.

7.5 Improving primal-dual methods for checking avoiding sure loss

The primal-dual method is the last one that we will investigate for checking avoiding sure loss. To start, we pose (P2) and (D2) in the suitable format of linear programs for the primal-dual method. We then discuss how we can improve this method by applying extra stopping criteria and a direct way to obtain feasible starting points. Note that this section closely follows [26, §4] and [27, §4.3].

Recall that the primal-dual method solves both primal and dual problems which

are in the form of:

$$\begin{array}{lll} \text{(P)} & \min\ c^\top x & \text{(D)} \quad \max\ b^\top y \\ & \text{s.t. } Ax = b, & \text{s.t. } A^\top y + t = c, \\ & x \geq 0 & t \geq 0,\ y \text{ free.} \end{array}$$

The problem (P3) is already in the form of (P) and its dual is:

$$\begin{aligned} \text{(D5)} \qquad & \max \quad 0 && \text{(D5a)} \\ & \text{subject to} \quad \forall f_i \in \mathcal{D}\colon \sum_{\omega \neq \omega_0} (f_i(\omega) - f_i(\omega_0))v(\omega) + t_i = f_i(\omega_0) && \text{(D5b)} \\ & \qquad\qquad q - \sum_{\omega \neq \omega_0} v(\omega) = 1 && \text{(D5c)} \\ & \qquad\qquad \forall \omega \neq \omega_0\colon v(\omega) + p(\omega) = 0 && \text{(D5d)} \\ & \qquad\qquad \forall i\colon t_i \geq 0,\ \forall \omega \neq \omega_0\colon p(\omega) \geq 0 \text{ and } q \geq 0. && \text{(D5e)} \end{aligned}$$

Because $-v(\omega) = p(\omega) \geq 0$ for all ω, (D5) is equivalent to (D4), as expected. The primal-dual method solves (P3) and (D5) simultaneously. Note that the primal-dual method can start with any an arbitrary point (x, y, t) where $x > 0$ and $t > 0$. Fortunately, for (P3), we can apply theorem 43 to obtain an initial interior feasible point. However, there is no closed form feasible point for (D5) (if we had, then we immediately would have found an optimal solution). In this case, a starting point of (D5) can be $q^0 = p^0(\omega) = 1/|\Omega|$, $v^0(\omega) = -1/|\Omega|$ for all ω, and $t_i^0 = 1$ for all i.

Remember that we can apply lemma 9 to (P3) only if we can keep all iterative points in the feasible region. Even though (P3) starts with a feasible point, the next iterative points do not necessarily stay in the feasible region because of numerical rounding errors. Therefore, it is good practice to calculate the primal residual and only apply lemma 9 if this error is negligible.

Now, consider solving the problem (D4') by the primal-dual method. In this

case, the dual of (D4') can be written in the form of (D) as follows:

$$\begin{aligned}
\text{(P4')} \qquad \max \quad & \sum_{i=1}^{n} \lambda_i f_i(\omega_0) + \alpha && \text{(P4'a)}\\
\text{subject to} \quad & \forall \omega \neq \omega_0 \colon \sum_{i=1}^{n} (f_i(\omega_0) - f_i(\omega))\lambda_i + \alpha + s(\omega) = 0 && \text{(P4'b)}\\
& \forall i \colon \lambda_i + u_i = 0 && \text{(P4'c)}\\
& \alpha + \beta = 0 && \text{(P4'd)}\\
& \sum_{i=1}^{n} r_i \lambda_i + \mu = 1 && \text{(P4'e)}\\
& \forall i \colon u_i \geq 0,\ \forall \omega \neq \omega_0 \colon s(\omega) \geq 0,\ \beta \geq 0 \text{ and } \mu \geq 0. && \text{(P4'f)}
\end{aligned}$$

To find an initial interior feasible point of (P4'), we first choose $\lambda_i < 0$ such that $\sum_{i=1}^{n} r_i \lambda_i < 1$. The $u_i > 0$ are then fixed by eq. (P4'c), and μ is fixed by eq. (P4'e). Note that $\mu = 1 - \sum_{i=1}^{n} r_i \lambda_i > 0$ by construction. Substituting $\alpha = -\beta$ into eq. (P4'b), we can then apply theorem 43 to find interior feasible values for β and $s(\omega)$ for all $\omega \neq \omega_0$. Unfortunately we cannot apply lemma 9 to (P4'), because the problem is no longer fully degenerate.

To sum up, for checking avoiding sure loss by the primal-dual method, the method solves either a pair of (P3) and (D5), or a pair of (P4') and (D4'). For each case, we have closed forms for obtaining initial feasible points for the primal problems, i.e., (P3) and (P4'). By exploiting an initial feasible point, lemma 9 can be applied to (P3) as the problem is fully degenerate, but lemma 9 can not be applied to (P4') as it is not fully degenerate.

7.6 Summary

In this chapter, we studied linear programs for checking avoiding sure loss and proposed our linear programs whose size are slightly reduced and have only non-negative

variables. We investigated how we can improve the simplex, the affine scaling and the primal-dual methods for efficiently solving our proposed linear programs.

To see an improvement, we will benchmark these three methods for checking avoiding sure loss. Therefore, in the next chapter, we will provide several algorithms for generating random sets of desirable gambles that either avoid or do not avoid sure loss. Then we will benchmark these methods on those randomly generated sets.

Chapter 8

Algorithms for generating sets of desirable gambles

This chapter is closely following [27, §6 and §7]. The aim of this chapter is to provide algorithms for generating arbitrary finite sets of gambles that either avoid or do not avoid sure loss. These arbitrary sets should be sufficiently generic so that we can use them for benchmarking algorithms for checking avoiding sure loss. We first give several algorithms for generating those coherent lower previsions that we mentioned in section 6.6. Next, we use these generated coherent lower previsions to construct sets of desirable gambles that either avoid or do not avoid sure loss.

8.1 Algorithms for generating coherent previsions

In this section, we give algorithms for generating coherent previsions, polyhedral lower previsions and linear-vacuous mixtures. All algorithms are presented in [27, §6]. To start, we first generate coherent previsions via an expectation operator.

Algorithm 1 Generate a coherent prevision [27, Algorithm 1]

Input: set of outcomes Ω

Output: a coherent prevision P on $\mathcal{L}(\Omega)$

1. Generate a probability mass function p as follows:
   (a) For each ω, sample r_ω uniformly from $(0, 1)$.
   (b) For each ω, set $p(\omega) \coloneqq \dfrac{\ln r_\omega}{\sum_{\omega\in\Omega} \ln r_\omega}$.
2. Generate a coherent prevision P
   (a) For any $f \in \mathcal{L}(\Omega)$, $P(f) \coloneqq E_p(f)$ as in eq. (6.21).

Note that the procedure in the state 1 in algorithm 1 gives $-\ln r_\omega$ an exponential Exp(1) distribution. If there are n outcomes, then this will give $p(\omega_1), \ldots, p(\omega_n)$ a Dirichlet Dir$(1, \ldots, 1)$ distribution [8, p.12]. This has uniform density over the unit simplex.

By algorithm 1, we can easily generate a finite set of probability mass functions which can be used to construct a polyhedral coherent lower prevision as in eq. (6.25). We present algorithm 2 for generating a polyhedral coherent lower prevision:

Algorithm 2 Generate a polyhedral lower prevision [27, Algorithm 2]

Input: a set of outcomes Ω and k coherent previsions: $Q_1, \ldots, Q_k$ (e.g. obtained by algorithm 1)

Output: a polyhedral lower prevision $\underline{P}$ on $\mathcal{L}(\Omega)$

1. For any $f \in \mathcal{L}(\Omega)$, $\underline{P}(f) \coloneqq \min_{j=1}^{k}\{Q_j(f)\}$.

An algorithm for generating a linear-vacuous mixture is given as follows:

Algorithm 3 Generate a linear-vacuous mixture [27, Algorithm 3]

Input: a set of outcomes Ω, coherent prevision Q (e.g. generated by algorithm 1) and $\delta \in (0, 1)$ (e.g. sample δ uniformly from $(0, 1)$),

Output: a linear-vacuous mixture $\underline{P}$

1. For any $f \in \mathcal{L}(\Omega)$, $\underline{P}(f) \coloneqq (1 - \delta)Q(f) + \delta \inf f$.

We now have three options to generate coherent lower previsions $\underline{E}$, namely, through (i) coherent previsions (algorithm 1), (ii) polyhedral lower previsions (al-

gorithm 2) and (iii) linear-vacuous mixtures (algorithm 3). Next, we will use these coherent lower previsions to generate arbitrary sets of desirable gambles.

8.2 Generating sets of desirable gambles that avoid sure loss

Given coherent lower previsions, we explain how to generate an arbitrary set of desirable gambles that avoids sure loss. As we will see later, if this coherent lower prevision is more generic, then the generated set of desirable gambles will also be more generic.

We start by generating a coherent lower prevision $\underline{E}$ on $\mathcal{L}(\Omega)$ e.g. through one of the above algorithms. After that, we generate a finite subset of gambles $\mathcal{K}$ of $\mathcal{L}(\Omega)$ and set $\underline{P}(f) := \underline{E}(f)$ for all $f \in \mathcal{K}$. Note that $\underline{P}$ is also coherent because it is the restriction of a coherent lower prevision [39, p. 58]. Therefore, the set $\mathcal{D} := \{f - \underline{P}(f) \colon f \in \mathcal{K}\}$ avoids sure loss. We summarised this idea and presented the following algorithm in [27, Algorithm 4]:

Algorithm 4 Generate a set of desirable gambles that avoids sure loss

---

Input: a set of outcomes Ω, a number of desirable gambles $n := |\mathcal{D}|$ and a coherent lower prevision $\underline{E}$ on $\mathcal{L}(\Omega)$

Output: a finite set of desirable gambles $\mathcal{D}$ that avoids sure loss

1. Generate $\{f_j \colon j \in \{1, \ldots, n\}\}$: for each ω and j, sample $f_j(\omega)$ uniformly from $(0, 1)$.
2. For each $i \in \{1, \ldots, n\}$, calculate $\underline{E}(f_i)$.
3. Set $\mathcal{D} := \{f_i - \underline{E}(f_i) \colon i \in \{1, \ldots, n\}\}$.

An overview of the logical structure of generating coherent lower previsions and arbitrary sets of desirable gambles is summarised in the diagram of fig. 8.1.

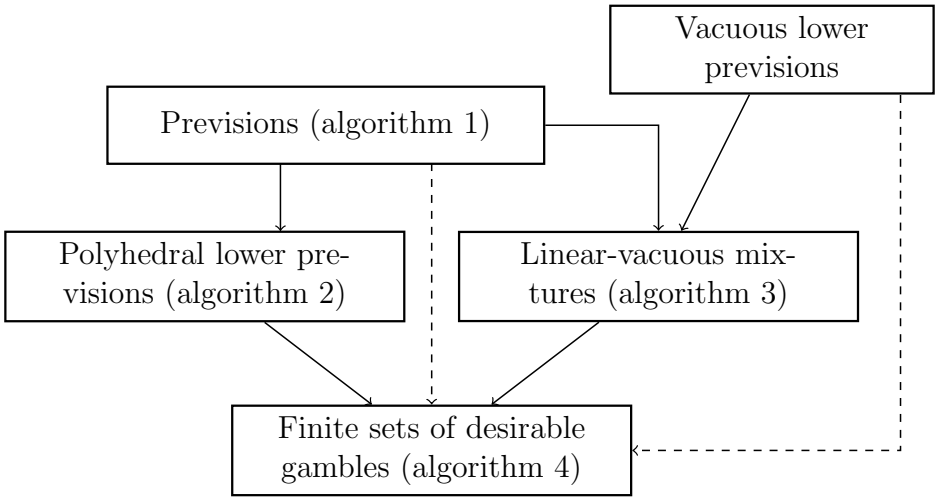

Figure 8.1: A diagram of generating coherent lower previsions and finite sets of desirable gambles that avoid sure loss.

Next, we investigate which type of coherent lower prevision we should use to generate sets of desirable gambles that avoid sure loss. This following work is presented in [27, §5]. To answer this question, we first look at $\mathcal{M}_{\underline{E}}$ and its extreme points, for various classes of $\underline{E}$:

(i) Vacuous lower prevision: $\mathcal{M}_{\underline{E}} = \Delta(\Omega)$ and its extreme points are all $0-1$ valued probabilities.

(ii) Coherent previsions: $\mathcal{M}_{\underline{E}} = \operatorname{ext} \mathcal{M}_{\underline{E}} = \{p\}$, $p \in \Delta(\Omega)$.

(iii) Polyhedral lower previsions: as in eq. (6.25), when M is finite, $\mathcal{M}_{\underline{E}}$ is a polyhedron and has a finite set of extreme points.

(iv) Linear-vacuous mixtures: for $p_0 \in \Delta(\Omega)$ and $\delta \in (0,1)$, $\mathcal{M}_{\underline{E}} = \{(1-\delta)p_0 + \delta p, p \in \Delta(\Omega)\}$ and $\operatorname{ext} \mathcal{M}_{\underline{E}} = \{(1-\delta)p_0 + \delta p,\ p \text{ is a } 0-1 \text{ valued probability}\}$.

Figure 8.2 illustrates examples of $\mathcal{M}_{\underline{E}}$ associated with different coherent lower previsions. For vacuous lower previsions and prevision, the number of extreme points are trivial. For polyhedral lower previsions, the number of extreme points is arbitrary (but finite). For linear-vacuous mixtures, the number of extreme points is limited to the number of outcomes, and the shape of its credal set is fixed (up

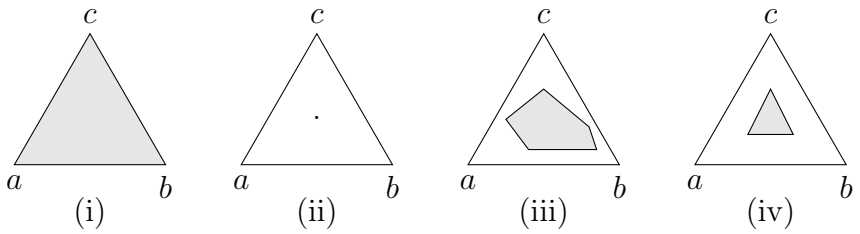

Figure 8.2: Simplex representation of different $\mathcal{M}_{\underline{E}}$ for $\Omega = \{a, b, c\}$: (i) vacuous, (ii) prevision, (iii) polyhedral lower prevision, (iv) linear-vacuous mixture. Reprinted from [27, Figure 1].

to scale and translation). Therefore, we do not recommend to use vacuous lower previsions or previsions to generate finite sets of desirable gambles as indicated by dashed lines in fig. 8.1.

Suppose that $\mathcal{D} = \{f - \underline{P}(f) \colon f \in \operatorname{dom} \underline{P}\}$ is generated by algorithm 4. Its credal set associated with $\mathcal{D}$ is given by

$$\mathcal{M}_{\mathcal{D}} = \{p \in \Delta(\Omega) \colon \forall f \in \mathcal{D},\ E_p(f) \geq 0\}. \tag{8.1}$$

We show that $\mathcal{M}_{\mathcal{D}}$ and $\mathcal{M}_{\underline{E}}$ are related as follows:

Corollary 44. [27, Corollary 1] Let $\underline{E}$ be a coherent lower prevision on $\mathcal{L}(\Omega)$, let $\underline{P}$ be a restriction of $\underline{E}$ to a finite domain, and let $\mathcal{D} = \{f - \underline{P}(f) \colon f \in \operatorname{dom} \underline{P}\}$. Then:

(i) $\mathcal{M}_{\underline{E}} \subseteq \mathcal{M}_{\mathcal{D}}$.

(ii) If $\mathcal{M}_{\underline{E}}$ is a polyhedron, then there is a finite set $\mathcal{K} \subseteq \mathcal{L}(\Omega)$ such that if $\operatorname{dom} \underline{P} = \mathcal{K}$, then $\mathcal{M}_{\mathcal{D}} = \mathcal{M}_{\underline{E}}$.

□

Proof. (i). We find that

$$\mathcal{M}_{\underline{E}} = \bigcap_{f\in\mathcal{L}(\Omega)} \{p \in \Delta(\Omega)\colon E_p(f) \geq \underline{E}(f)\} \tag{8.2}$$

$$= \bigcap_{f\in\mathcal{L}(\Omega)} \{p \in \Delta(\Omega)\colon E_p(f - \underline{E}(f)) \geq 0\} \tag{8.3}$$

$$\subseteq \bigcap_{f\in\operatorname{dom}\underline{P}} \{p \in \Delta(\Omega)\colon E_p(f - \underline{P}(f)) \geq 0\} = \mathcal{M}_{\mathcal{D}}. \tag{8.4}$$

(ii). By (i), we only need to show that $\mathcal{M}_{\mathcal{D}} \subseteq \mathcal{M}_{\underline{E}}$. Since $\mathcal{M}_{\underline{E}}$ is a polyhedron, it is an intersection of a finite number of half-spaces. Therefore, there exists an $n \in \mathbb{N}$, vectors $f_1, \ldots, f_n$ and numbers $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ such that

$$\mathcal{M}_{\underline{E}} = \bigcap_{i=1}^{n} \{p \in \Delta(\Omega)\colon p \cdot f_i \geq \alpha_i\} \tag{8.5}$$

where '$\cdot$' denotes the dot product. Note that

$$\underline{E}(f_i) = \min_{p\in\mathcal{M}_{\underline{E}}} p \cdot f_i = \min_{p} \{p \cdot f_i \colon \forall j,\ p \cdot f_j \geq \alpha_j\} \geq \alpha_i. \tag{8.6}$$

Let $\mathcal{K} = \{f_1, \ldots, f_n\}$ and $\operatorname{dom}\underline{P} = \mathcal{K}$. Then,

$$\mathcal{M}_{\mathcal{D}} = \bigcap_{i=1}^{n} \{p \in \Delta(\Omega)\colon p \cdot f_i \geq \underline{E}(f_i)\} \tag{8.7}$$

$$\subseteq \bigcap_{i=1}^{n} \{p \in \Delta(\Omega)\colon p \cdot f_i \geq \alpha_i\} = \mathcal{M}_{\underline{E}}. \tag{8.8}$$

□

Therefore, if we want $\mathcal{M}_{\mathcal{D}}$ to have a sufficient number of extreme points so that $\mathcal{D}$ will be generic for benchmarking, then we should generate $\mathcal{D}$ by using either a polyhedral lower prevision, or at the very least using a linear-vacuous mixture [27].

Figure 8.3 visualises the construction of the proof of corollary 44, for various cases of lower previsions. Corollary 44 implies that we do not necessarily achieve new extreme points even though we add more and more gambles to $\mathcal{D}$. Therefore, if

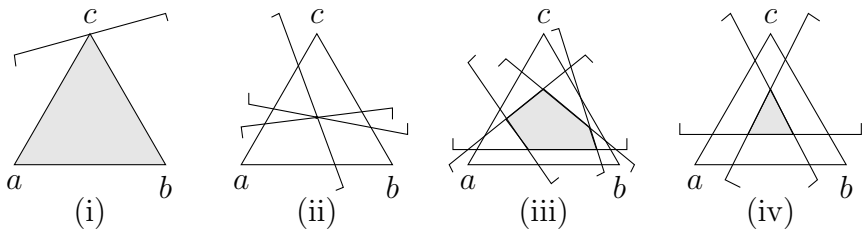

Figure 8.3: Constructing $\mathcal{M}_{\underline{E}}$ by finite half-spaces for $\Omega = \{a, b, c\}$: (i) vacuous, (ii) prevision, (iii) polyhedral lower prevision, (iv) linear-vacuous mixture. Reprinted from [27, Figure 2].

we want to gain new extreme points with every gamble that we add, then algorithm 4 may not be a good choice for generating $\mathcal{D}$ [27]. In the next section, we suggest another algorithm that precisely addresses this issue.

8.3 Sequentially generating sets that do not avoid sure loss

This whole section closely follows [27, §6.1]. Suppose that we have a set $\mathcal{E} = \{f_1, \ldots, f_n\}$ that avoids sure loss (which can be obtained by algorithm 4), or we can also start with an empty set which also avoids sure loss. Suppose that we want to find a gamble g for which $\mathcal{E} \cup \{g\}$ avoids sure loss. However, it is not easy to generate g as we want. Indeed, we can shift g by α such that $g - \alpha$ satisfies this condition. To do so, we first find the range of values for α such that $\mathcal{E} \cup \{g - \alpha\}$ avoids sure loss.

By the condition of avoiding sure loss in definition 18, a constraint on the values of α must satisfy the following constraint: for all $n \in \mathbb{N}$, all $\lambda_1, \ldots, \lambda_n \geq 0$, and all $f_1, \ldots, f_n \in \mathcal{E}$,

$$\sup_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i f_i(\omega) + g(\omega) \right) \geq \alpha. \tag{8.9}$$

The infimum of this upper bound is precisely $\overline{E}_{\mathcal{E}}(g)$, which can be obtained by solving a linear programming problem ((Q1) in algorithm 5). So, we proved:

Corollary 45. [27, Corollary 2] Let $\mathcal{E}$ avoid sure loss and let $g \in \mathcal{L}(\Omega)$. $\mathcal{E} \cup \{g - \alpha\}$ avoids sure loss if and only if $\alpha \leq \overline{E}_{\mathcal{E}}(g)$. □

Hence, if we set $\alpha > \overline{E}_{\mathcal{E}}(g)$, then $\mathcal{E} \cup \{g - \alpha\}$ does not avoid sure loss. We presented the following algorithm for generating a set of desirable gambles that does not avoid sure loss in [27, Algorithm 5].

Algorithm 5 Generate a set of desirable gambles that does not avoid sure loss

Input: a set of outcomes Ω, a set of desirable gambles $\mathcal{E}$ that avoids sure loss and $\delta > 0$ (e.g. sample δ uniformly from $(0, 1)$)

Output: a set of desirable gambles $\mathcal{D}$ that does not avoid sure loss.

1. For each $\omega \in \Omega$, sample $g(\omega)$ uniformly from $(0, 1)$.
2. Calculate $\overline{E}_{\mathcal{E}}(g)$ by solving the following linear program:

$$\text{(Q1)} \quad \min \quad \beta \tag{Q1a}$$

$$\text{s. t.} \quad \forall \omega \in \Omega\colon \sum_{f_i \in \mathcal{E}} f_i(\omega)\lambda_i - \beta \leq -g(\omega), \quad \lambda_i \geq 0 \quad (\beta \text{ free}). \tag{Q1b}$$

3. Set $\mathcal{D} \coloneqq \mathcal{E} \cup \{g - \overline{E}_{\mathcal{E}}(g) - \delta\}$.

Note that since there is only a single gamble in sets of gambles generated by algorithm 5 that violates avoiding sure loss, these sets are the most computationally challenging sets to identify not avoiding sure loss. Consequently, they are the most suitable sets for benchmarking, as any measurable improvement on these sets implies an at least as large improvement on any simpler scenarios.

Next, we provide an algorithm for generating sets of desirable gambles that still avoid sure loss. The next section closely follows [27, §6.2]

8.4 Sequentially generating sets that avoid sure loss

Consider a set $\mathcal{E} := \{f - \underline{Q}(f) \colon f \in \operatorname{dom} \underline{Q}\}$ that avoids sure loss where $\underline{Q}$ is a coherent lower prevision. Let g be a gamble in $\mathcal{L}(\Omega) \setminus \operatorname{dom} \underline{Q}$. By corollary 45, we know that the larger set $\mathcal{D} := \mathcal{E} \cup \{g - \alpha\}$ still avoids sure loss as long as $\alpha \leq \overline{E}_{\mathcal{E}}(g)$. Note that the number of extreme points of $\mathcal{M}_{\mathcal{D}}$ can be less than $\mathcal{M}_{\mathcal{E}}$ after adding $\{g - \alpha\}$, as shown in fig. 8.4.

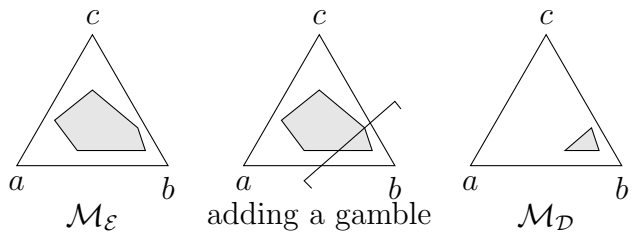

Figure 8.4: Simplex representation of a credal sets after adding a gamble. Reprinted from [27, Figure 3].

How can we keep the number of extreme points after adding $\{g - \alpha\}$? If $\underline{P}$ is a coherent extension of $\underline{Q}$ to $\operatorname{dom} \underline{Q} \cup \{g\}$, then $\mathcal{M}_{\underline{P}}$ must have at least as many extreme points as $\mathcal{M}_{\underline{Q}}$. This is because by coherence, for every $f \in \operatorname{dom} \underline{Q}$, $\underline{P}(f)$ and $\underline{Q}(f)$ must be achieved at some extreme point of $\mathcal{M}_{\underline{P}}$ and $\mathcal{M}_{\underline{Q}}$, respectively [46, p. 126]. However, since $\underline{P}(f) = \underline{Q}(f)$ for all these gambles f, $\mathcal{M}_{\underline{P}}$ cannot have fewer extreme points than $\mathcal{M}_{\underline{Q}}$, because otherwise $\underline{P}(f) > \underline{Q}(f)$ for at least one gamble f. Consequently, the number of extreme points does not decrease if we can preserve coherence.

Theorem 46. (adapted from [46, p. 126]) Let $\underline{Q}$ be a coherent lower prevision and let g be a gamble in $\mathcal{L}(\Omega) \setminus \operatorname{dom} \underline{Q}$. Suppose that $\underline{P}$ is an extension of $\underline{Q}$ to

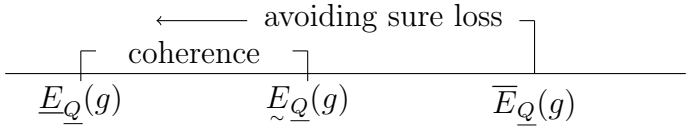

Figure 8.5: Ranges of avoiding sure loss and coherence. Reprinted from [27, Figure 4].

$\operatorname{dom} \underline{Q} \cup \{g\}$. Then $\underline{P}$ is coherent if and only if $\underline{P}(g) \in [\underline{E}_{\underline{Q}}(g), \underline{\underline{E}}_{\underline{Q}}(g)]$, where

$$\underline{\underline{E}}_{\underline{Q}}(g) := \inf_{\substack{f_i \in \operatorname{dom} \underline{Q}, \\ \lambda_i \geq 0}} \left\{ \max_{\omega \in \Omega} \left(g(\omega) + \sum_{i=1}^{n} \lambda_i (f_i(\omega) - \underline{Q}(f_i))) - \lambda_0 (f_0(\omega) - \underline{Q}(f_0)) \right) \right\}. \tag{8.10}$$

□

Given a coherent lower prevision $\underline{Q}$, fig. 8.5 illustrates ranges of avoiding sure loss and coherence of g.

We can easily calculate $\underline{E}_{\underline{Q}}(g)$ by solving a single linear programming problem ((Q4) in algorithm 6). However, for $\underline{\underline{E}}_{\underline{Q}}(g)$, we first have to solve a separate linear programming problem for every $f_j \in \operatorname{dom} \underline{Q}$:

$$\text{(Q2)} \quad \min \quad \beta \tag{Q2a}$$

$$\text{s.t.} \quad \forall \omega \in \Omega : \sum_{i=1,\ i \neq j}^{n} \lambda_i (f_i(\omega) - \underline{Q}(f_i)) - \lambda_j (f_j(\omega) - \underline{Q}(f_j)) - \beta \leq -g(\omega) \tag{Q2b}$$

$$\forall i : \lambda_i \geq 0 \quad (\beta \text{ free}). \tag{Q2c}$$

Next, $\underline{\underline{E}}_{\underline{Q}}(g)$ is obtained by minimising the optimal values among these linear programs (Q2). Also note that each separate linear program is very similar to the linear program for calculating $\overline{E}_{\underline{Q}}(g)$ ((Q3) in algorithm 6) and has the same size. So, instead of evaluating $\underline{\underline{E}}_{\underline{Q}}(g)$, we prefer to calculate $\underline{E}_{\underline{Q}}(g)$ and $\overline{E}_{\underline{Q}}(g)$ (or $\overline{E}_{\mathcal{E}}(g)$ where $\mathcal{E} = \{f - \underline{Q}(f) \colon f \in \operatorname{dom} \underline{Q}\}$), because for each of them, we solve only one linear programming problem. Next, if we choose a very small number $\delta \in (0, 1)$ and set $\underline{P}(g) := (1 - \delta)\underline{E}_{\underline{Q}}(g) + \delta \overline{E}_{\underline{Q}}(g)$, then $\underline{P}(g)$ is slightly larger than $\underline{E}_{\underline{Q}}(g)$,

nevertheless it is less than $\underset{\sim}{E}_{\underline{Q}}(g)$. Consequently, $\mathcal{E} \cup \{g - \underline{P}(g)\}$ still avoids sure loss as we desire. We presented this approach in [27, Algorithm 6].

Algorithm 6 Sequentially generate a set of desirable gambles that avoids sure loss

Input: a set of outcomes Ω, a lower prevision $\underline{Q}$ that avoids sure loss (e.g. Algorithms 1, 2 or 3) a set $\mathcal{E} = \{f - \underline{Q}(f) \colon f \in \operatorname{dom} \underline{Q}\}$ that avoids sure loss and $\delta \in (0, 1)$.

Output: a larger set of desirable gambles $\mathcal{D}$ that avoids sure loss.

1. For each $\omega \in \Omega$, sample $g(\omega)$ uniformly from $(0, 1)$.
2. Calculate $\overline{E}_{\underline{Q}}(g)$ by solving

$$\begin{aligned} (\text{Q3}) \quad \min \quad & \beta && \text{(Q3a)} \\ \text{subject to} \quad & \forall \omega \in \Omega \colon \sum_{i=1}^{n} (f_i(\omega) - \underline{Q}(f_i))\lambda_i - \beta \leq -g(\omega) && \text{(Q3b)} \\ & \forall i : \lambda_i \geq 0,\ f_i(\omega) - \underline{Q}(f_i) \in \mathcal{E} \qquad (\beta \text{ free}). && \text{(Q3c)} \end{aligned}$$

3. Calculate $\underline{E}_{\underline{Q}}(g)$ by solving

$$\begin{aligned} (\text{Q4}) \quad \max \quad & \gamma && \text{(Q4a)} \\ \text{subject to} \quad & \forall \omega \in \Omega \colon \sum_{i=1}^{n} (f_i(\omega) - \underline{Q}(f_i))\lambda_i + \gamma \leq g(\omega) && \text{(Q4a)} \\ & \forall i : \lambda_i \geq 0,\ f_i(\omega) - \underline{Q}(f_i) \in \mathcal{E} \qquad (\gamma \text{ free}). && \text{(Q4a)} \end{aligned}$$

4. Set $\underline{P}(g) \coloneqq (1 - \delta)\underline{E}_{\underline{Q}}(g) + \delta \overline{E}_{\underline{Q}}(g)$.
5. Set $\mathcal{D} \coloneqq \mathcal{E} \cup \{g - \underline{P}(g)\}$.

Note that in our benchmarking, we will not use algorithm 6 to generate sets of desirable gambles that avoid sure loss. Instead, we use a combination of algorithm 2 and algorithm 4, which generates constraints via a set of probability mass functions, as this is computationally faster. We provide algorithm 6 for the sake of completeness, as an alternative algorithm for generating constraints directly.

8.5 Summary

To summarise, we gave several algorithms for generating coherent lower previsions, i.e., algorithm 1 for coherent previsions, algorithm 2 for polyhedral lower previsions and algorithm 3 for linear-vacuous mixtures. We then discussed how these lower previsions can be used to generate sets of desirable gambles that either avoid (algorithms 4 and 6) or do not avoid sure loss (algorithm 5) for benchmarking. In the next chapter, we will perform a simulation study and present benchmarking results.

Chapter 9

Benchmarking and discussion

In this simulation study, we compare the impact of those mentioned improvements for the simplex, the affine scaling and the primal-dual methods on randomly generated sets of desirable gambles that either avoid or do not avoid sure loss. We present benchmarking results and discuss an efficiency of our improved methods for checking avoiding sure loss. This chapter is based on [26, §7 and §8] and [27, §7 and §8]. Note that the plots in figs. 9.2 and 9.3 are presented in [26, §7 8] while the plots in figs. 9.1, 9.4, 9.5, 9.4, 9.5, 9.8, 9.9, 9.10 and 9.11, where we further run extra simulations comparing the impact of different improvements on the primal-dual methods.

9.1 Benchmarking results

To benchmark our theoretical improvements for these three methods, we first generate random sets of desirable gambles and then pose them as linear programming problems in the form that are suitable to different methods. Next, we solve these linear programs by our improved methods.

We generate sets of desirable gambles for benchmarking as follows. We generate

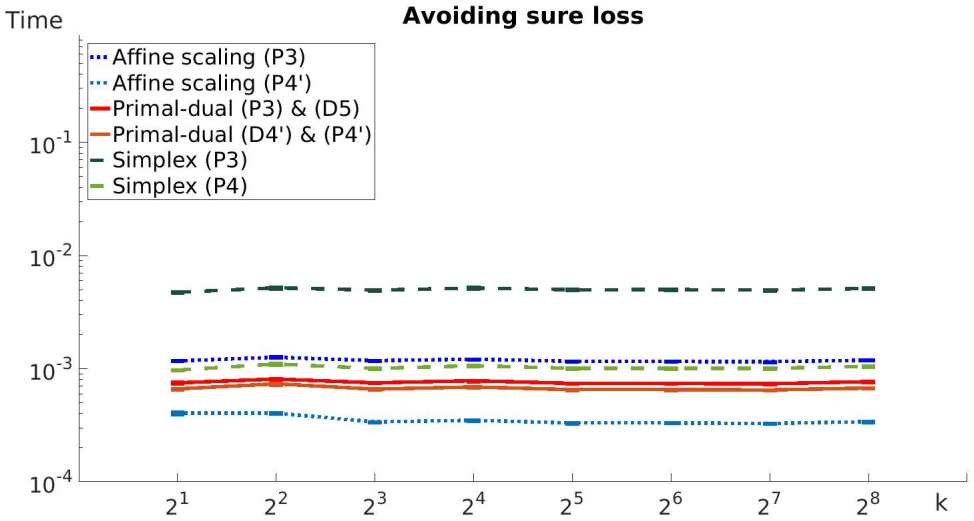

Figure 9.1: The average computational time for different methods for checking avoiding sure loss when varying k and fixed $|\mathcal{D}| = |\Omega| = 2^4$. All sets avoid sure loss

two cases of random sets: (i) sets that avoid sure loss, and (ii) sets that do not. For each case, we consider the size of sets of desirable gambles $\mathcal{D}$ for which $|\mathcal{D}| = 2^i$ for $i \in \{1, 2, \ldots, 8\}$ and the size of sets of outcomes Ω for which $|\Omega| = 2^j$ for $j \in \{1, 2, \ldots, 8\}$. Random sets of desirable gambles that avoid sure loss are generated as follows:

1. We use algorithm 1 to generate k coherent previsions. We fixed $k = 2^5$, since we observed that varying k has little impact on the results (see fig. 9.1).

2. From these k coherent previsions, we use algorithm 2 to generate a polyhedral lower prevision.

3. We use algorithm 4, with this polyhedral lower prevision, to generate a random set of desirable gambles $\mathcal{E}$ that avoids sure loss.

Next, starting from a set $\mathcal{E}$ that avoids sure loss, we generate a set that does not avoid sure loss using algorithm 5 with $\delta = 0.05$.

Linear programs	Methods		
	Simplex	Affine scaling	Primal-dual
(P3)	✓	✓	✓
(D3)	✓		
(D4')		✓	✓

Table 9.1: List of different methods for solving different linear programs for checking avoiding sure loss. Note that the primal-dual method also solves the dual problem simultaneously, i.e., (D5) for (P3) and (P4') for (D4') [27, Table 1].

For each random set of desirable gambles, we assume that we do not know whether it avoids sure loss or not. To find out, we pose the linear programming problem in the suitable format for each of the different methods. Table 9.1 gives an overview of these different algorithms solving different linear programming problems. Note that the primal-dual method simultaneously solves the primal and the dual problems.

To compare these three methods, we wrote our own implementation of the affine scaling and the primal-dual methods. We used an implementation of the revised simplex method written by Strang [37]. Recall that the revised simplex method is mathematically equivalent to the standard simplex method, but is much more efficient and numerically stable [7, §3.7]. We used the revised simplex method to solve both problems (P3) and (D3). For the affine scaling method, a standard version is used for solving (D4'), while an improved version, which implements the extra stopping criterion and our mechanism for calculating feasible starting points, is used to solve (P3). For the primal-dual method, we have two improved versions: (i) the first version that has the extra stopping criterion and the mechanism for calculating feasible starting points is used for solving (P3) and (D5), and (ii) the second version that has only the mechanism for calculating feasible starting points is used to solve (D4') and (P4').

For each method, we run the algorithm twice in order to remove any warm-up effects that may happen in the first run, and we only measure the corresponding

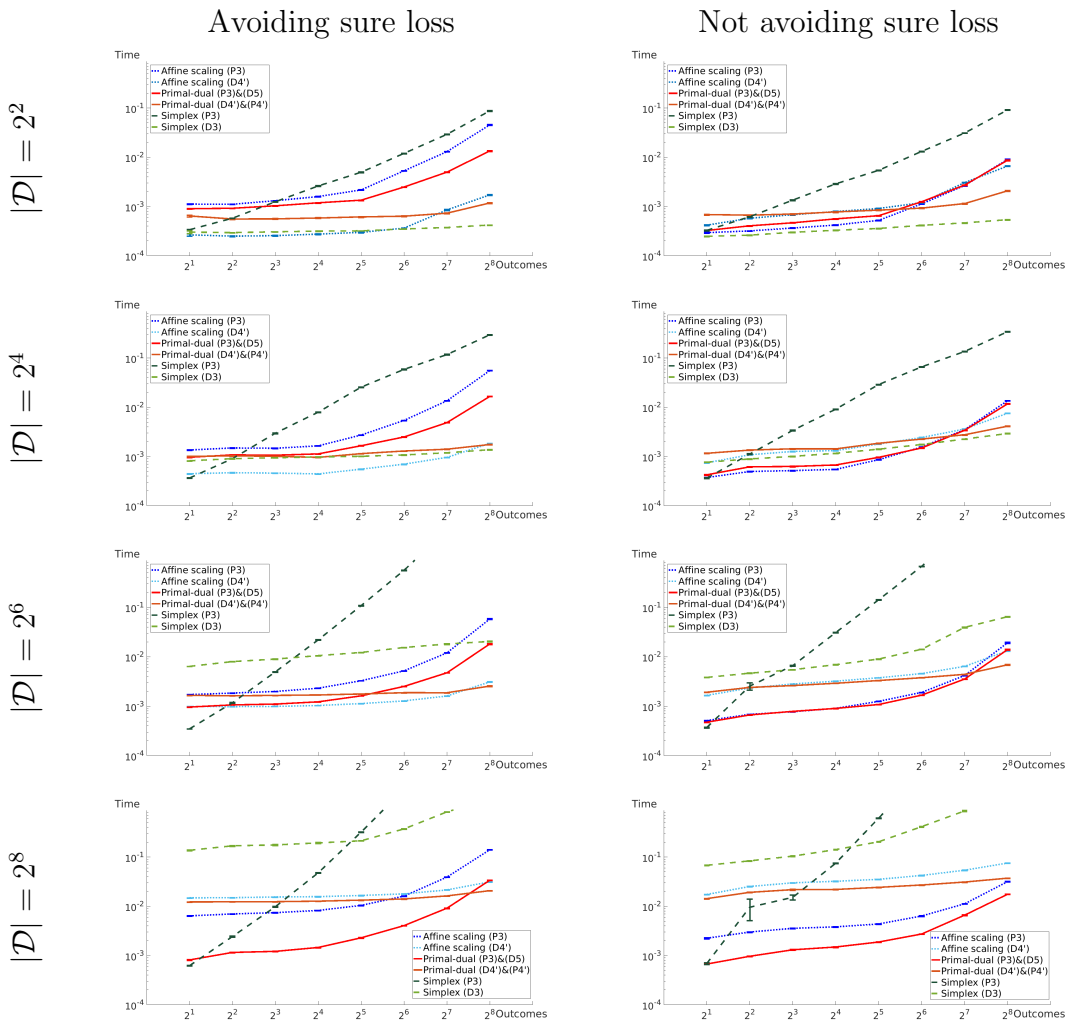

Figure 9.2: Comparison plots of the average computational time for three methods. The left column avoids sure loss and the right column does not. Each row represents a different number of desirable gambles where we vary the number of outcomes. The labels indicate different linear programs solved by different methods. Reprinted from [27, Figure 5]

computational time taken in the second run. The process is repeated 1000 times and a summary of the results is presented in figs. 9.2 and 9.3.

Figures 9.2 and 9.3 presents the average computational time taken during each method when solving different linear programs for checking avoiding sure loss. In the left column, the sets of desirable gambles avoid sure loss while in the right column, they do not avoid sure loss. In fig. 9.2 each row represents a different number of desirable gambles, and the horizontal axis represents the number of outcomes. In contrast, in fig. 9.3, each row represents a different number of outcomes, and the

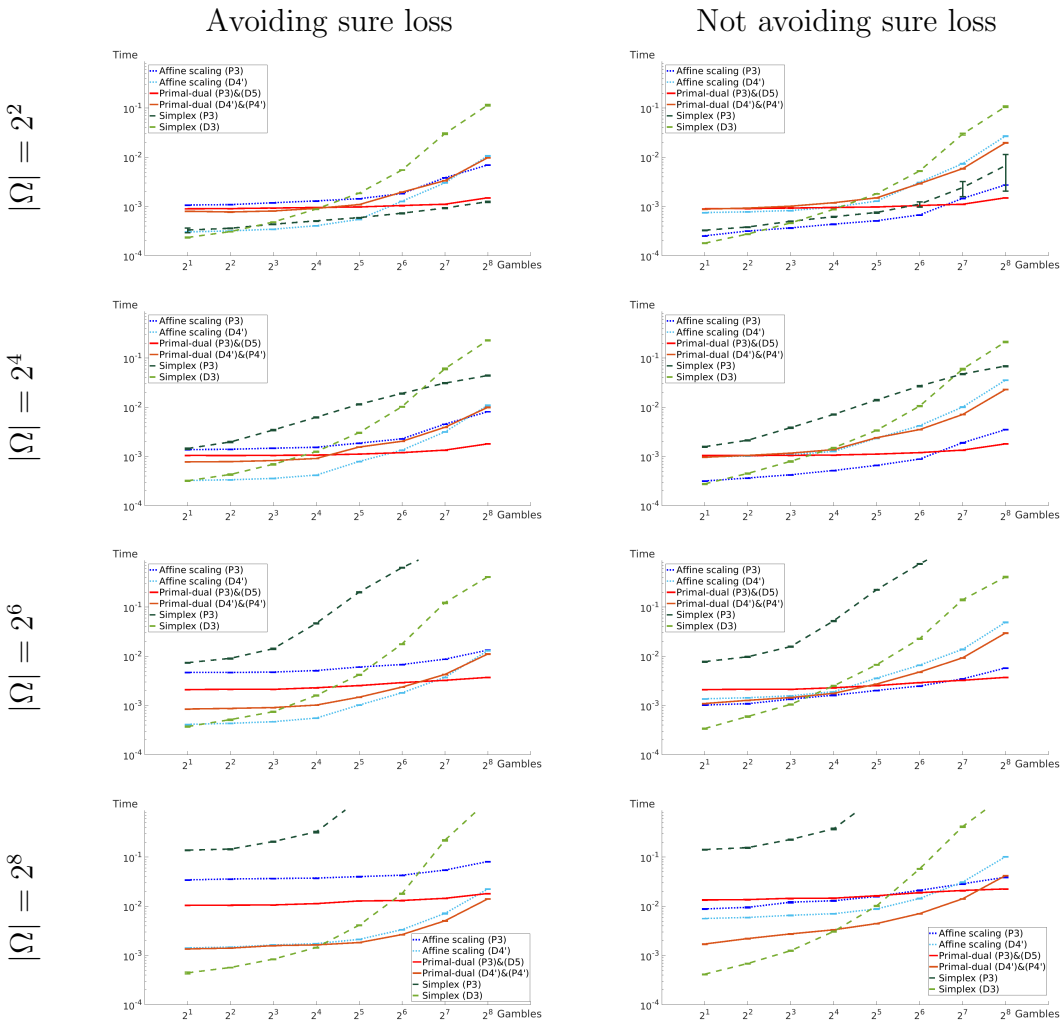

Figure 9.3: Comparison plots of the average computational time for three methods. The left column avoids sure loss and the right column does not. Each row represents a different number of outcomes where we vary the number of desirable gambles. The labels indicate linear programs solved by different methods. Reprinted from [27, Figure 6].

horizontal axis represents the number of desirable gambles. In both figures, the vertical axis shows the computational time which is averaged over 1000 random sets of desirable gambles. The error bars on the figures represent approximate 95% confidence intervals on the mean computation time. These bars are barely visible as the sample size is large, except in some rare cases where we observed large variability in the simplex method (possibly due to the numerical issues that we discussed earlier in section 2.4).

In the avoiding sure loss case, note that the sets of desirable gambles are always generated from coherent lower previsions. However, in some applied problems, this may not be the case. To address this point, we ran one further experiment where we added a negative bias in order to remove coherence whilst the sets still avoid sure loss. Specifically, in Stage 3 of algorithm 4, we set $\mathcal{D} \coloneqq \{f_i - \underline{E}(f_i) + \eta(f_i) \colon i \in \{1, \ldots, n\}\}$, for some $\eta(f_i) > 0$. For this experiment, we considered two scenarios: (i) we uniformly sampled each $\eta(f_i)$ from the open $(0, 1)$ interval, and (ii) we fixed each $\eta(f_i) \coloneqq 0.01$. This made no practical difference. In particular, the plots in figs. 9.2 and 9.3 for the avoiding sure loss case remained nearly identical, with no change in general conclusions.

In addition to presenting mean times and error bars from the simulation, we give information on the spread of times associated with some cases of these mean. Specifically, figs. 9.4 and 9.5 present box-and-whisker plots of the computational time for different algorithms for checking avoiding sure loss. Each box-and-whisker plot summarises the computational time for checking avoiding sure loss of 1000 random sets of desirable gambles. In the left column, the sets of desirable gambles avoid sure loss while in the right column, they do not avoid sure loss. In both figures, the horizontal axis represents different algorithms solving different linear programs. In fig. 9.4 each row represents a different number of desirable gambles, while, in fig. 9.5, each row represents a different number of outcomes.

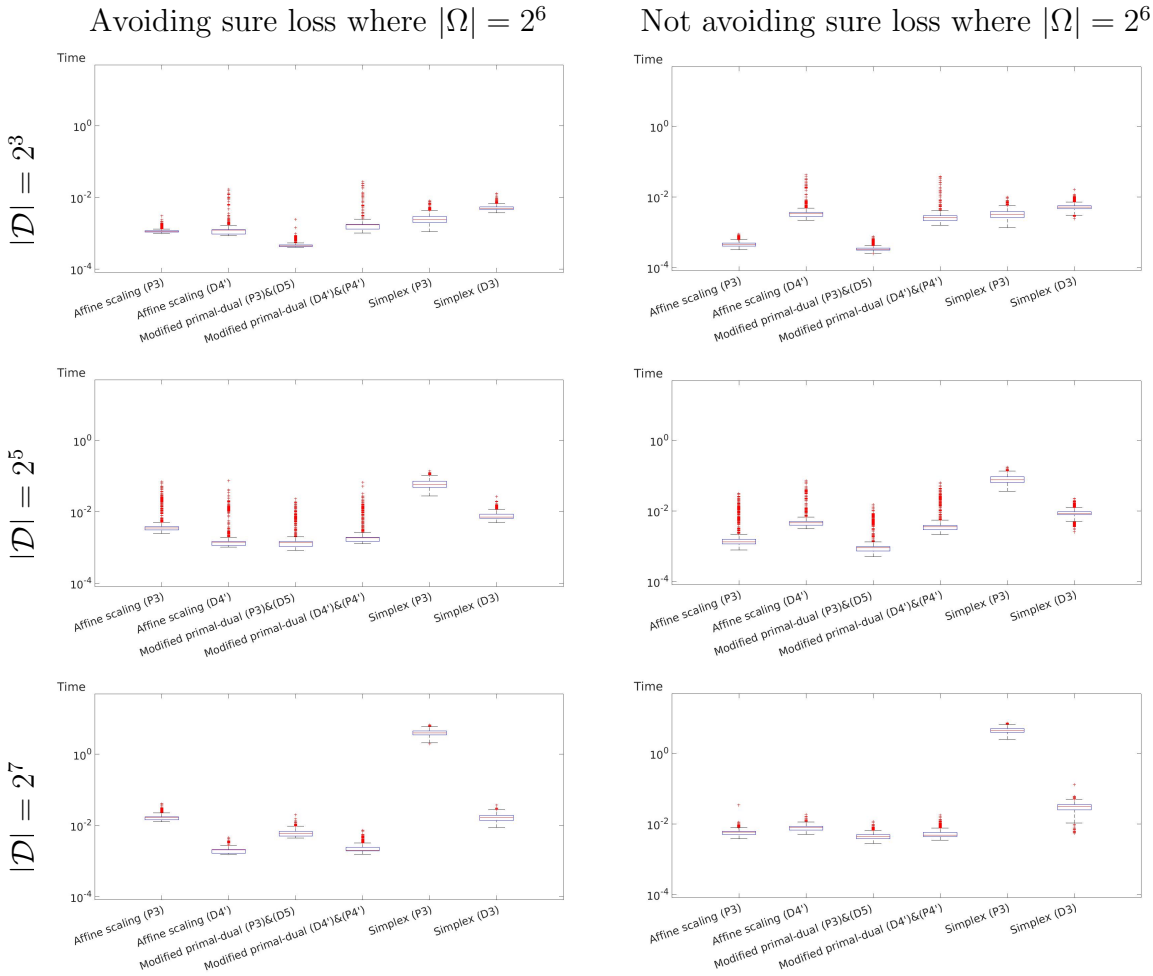

Figure 9.4: Comparison box-and-whisker plots of the computational time for three methods. The number of outcomes is $|\Omega| = 2^6$. The left column avoids sure loss and the right column does not. Each row represents a different number of desirable gambles. The horizontal axis represents different algorithms solving different linear programs. On each box, the central mark indicates the median, and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers extend to the most extreme data points not considered outliers, which are values that are more than three scaled median absolute deviations away from the median. The outliers are plotted individually using the '+' symbol.

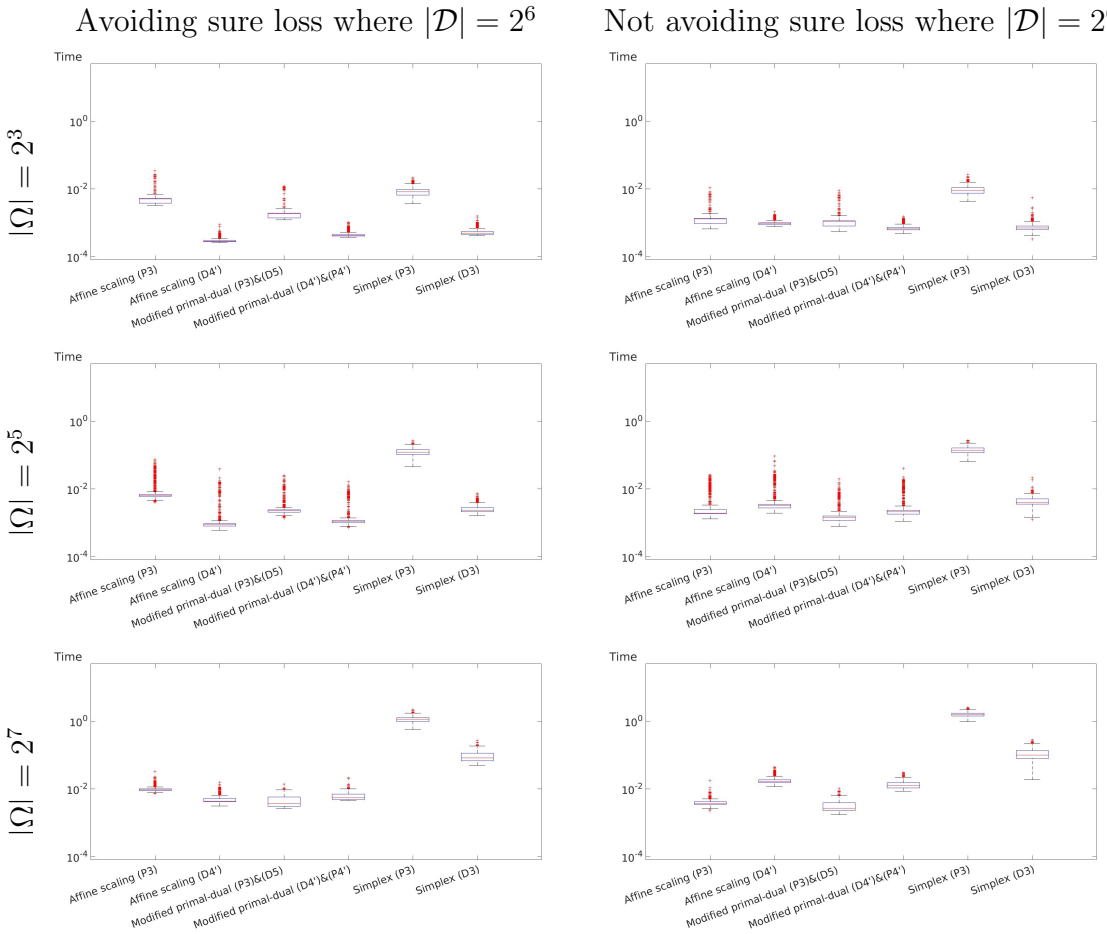

Figure 9.5: Comparison box-and-whisker plots of the computational time for three methods. The number of desirable gambles is $|\mathcal{D}| = 2^6$. The left column avoids sure loss and the right column does not. Each row represents a different number of outcomes. The horizontal axis represents different algorithms solving different linear programs. On each box, the central mark indicates the median, and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers extend to the most extreme data points not considered outliers, which are values that are more than three scaled median absolute deviations away from the median. The outliers are plotted individually using the '+' symbol.

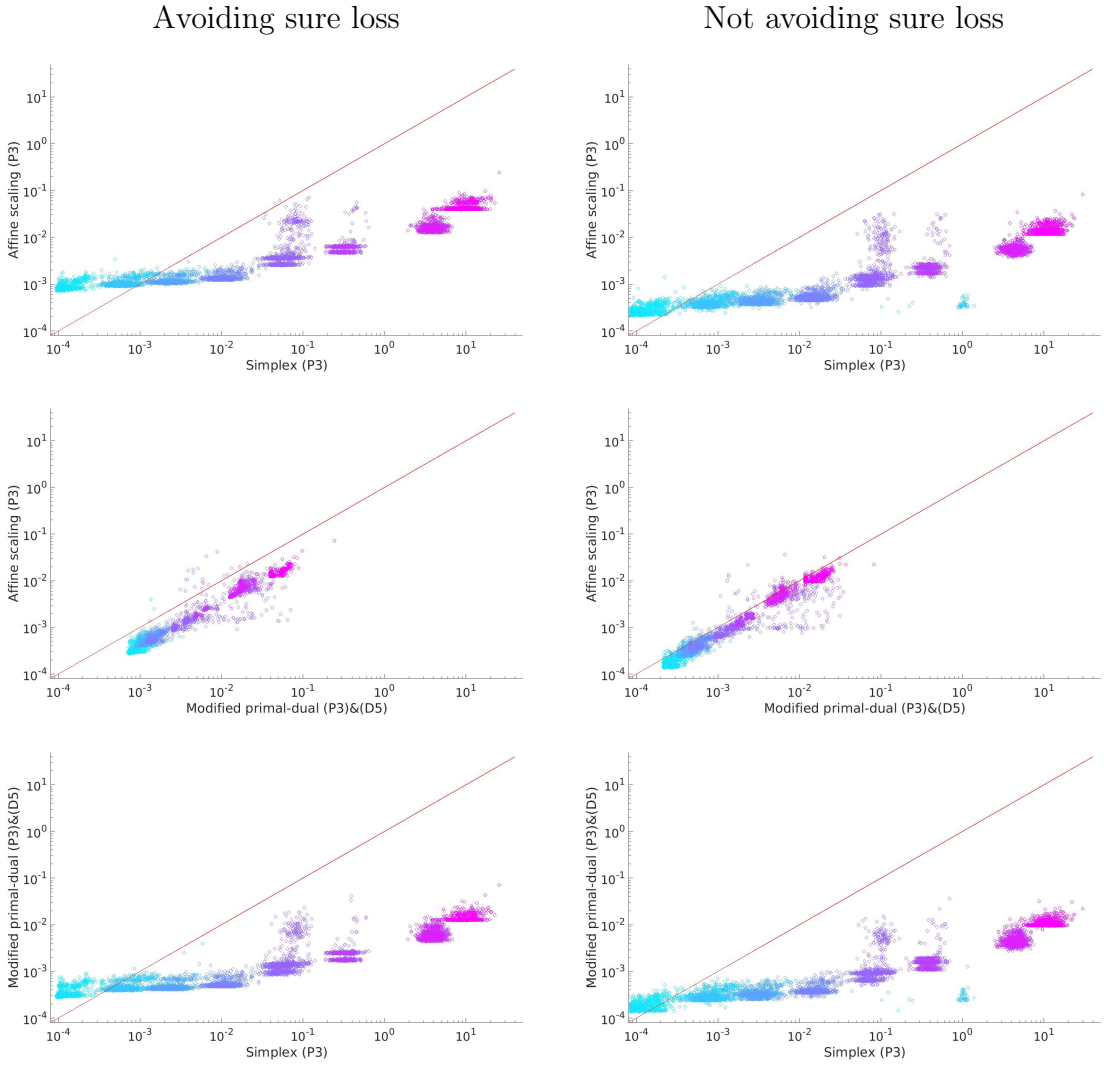

Figure 9.6: Comparison scatter plots of the computational time for different pairs of methods. The number of desirable gambles is $|\mathcal{D}| = 2^6$ where we vary numbers of outcomes for which $|\Omega| = 2^j$ for $j \in \{1, 2, \ldots, 8\}$. The left column avoids sure loss and the right column does not. Each axis shows computational time for each of different algorithms. The red line is the identity line.

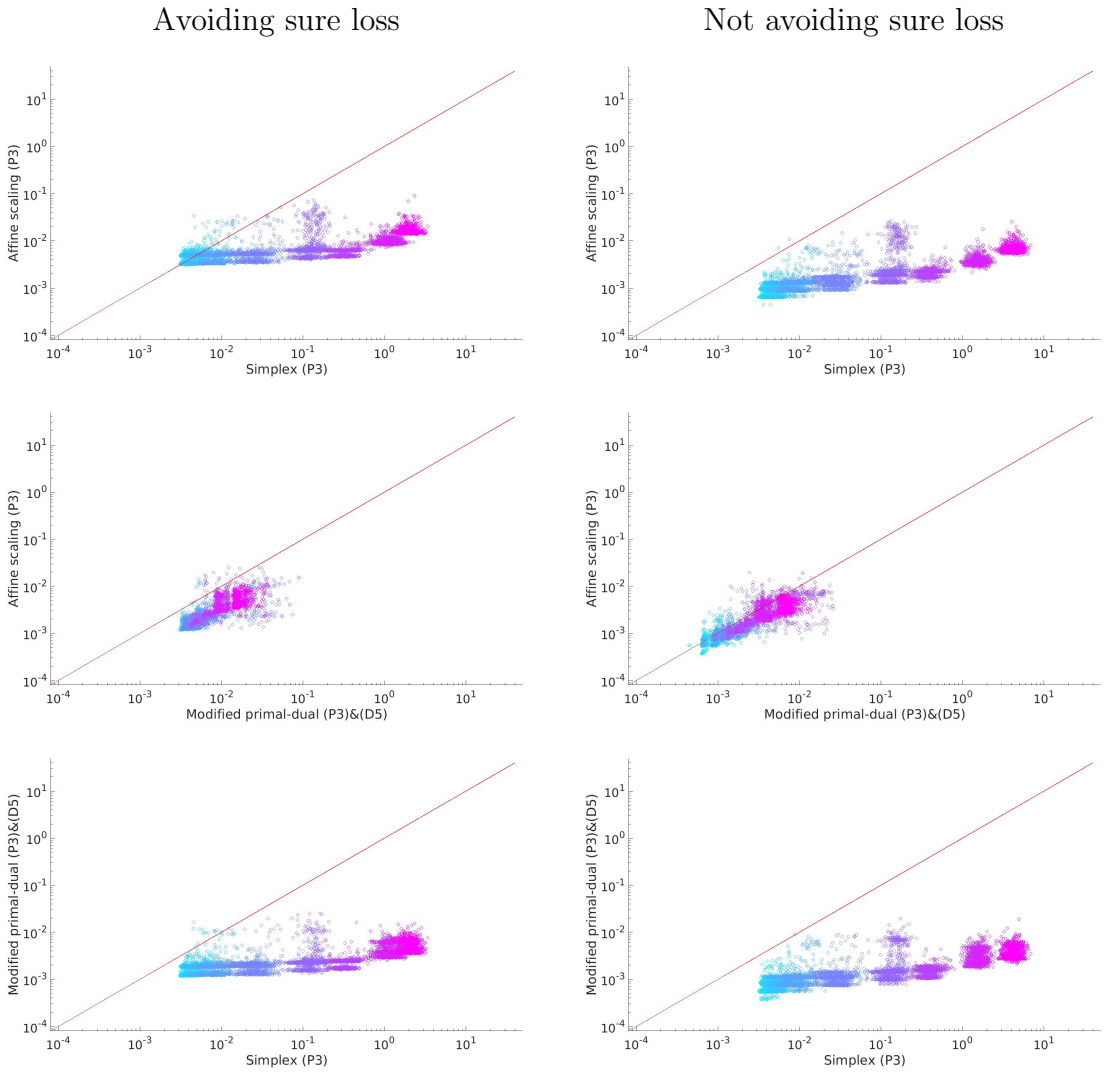

Figure 9.7: Comparison scatter plots of the computational time for different pairs of methods. The number of outcomes is $|\Omega| = 2^6$ where we vary numbers of desirable gambles for which $|\mathcal{D}| = 2^i$ for $i \in \{1, 2, \ldots, 8\}$. The left column avoids sure loss and the right column does not. Each axis shows computational time for each of different algorithms. The red line is the identity line.

We also present some scatter plots of the computational time for one method against the time for the other method for checking avoiding sure loss. This would show whether the computational times under the two methods are related to each other and perhaps whether there are distinct groups of simulations where one method has done better than the other.

Figures 9.6 and 9.7 present scatter plots of the computational time for different pairs of algorithms for checking avoiding sure loss. In the left column, the sets of desirable gambles avoid sure loss while in the right column, they do not avoid sure loss. In both figures, the horizontal and vertical axis represent the computational time of one method against the computational time for the other. In fig. 9.6 we fix the number of desirable gambles to be $|\mathcal{D}| = 2^6$ and vary numbers of outcomes for which $|\Omega| = 2^j$ for $j \in \{1, 2, \ldots, 8\}$. In fig. 9.7, we fix the number of outcomes to be $|\Omega| = 2^6$ and vary numbers of desirable gambles for which $|\mathcal{D}| = 2^i$ for $i \in \{1, 2, \ldots, 8\}$.

We also compare our suggested improvements on the primal-dual methods. Specifically, when we solve (P3) and (D5), we investigate an impact of implementations of extra stopping criterion and feasible starting points in the primal-dual method. For solving (D4') and (P4'), we only investigate an impact of an implementation of feasible starting points.

Figures 9.8, 9.9, 9.10 and 9.11 present the average computational time for different improved primal-dual methods for solving different linear programs. Specifically, we fix the number of desirable gambles and change the number of outcomes in figs. 9.8 and 9.9 while in figs. 9.10 and 9.11, we fix the number of outcomes and change the number of desirable gambles. In these figures, the sets of desirable gambles in the left column avoid sure loss while in the right column, they do not avoid sure loss. The vertical axis represents the computational time which is averaged over 1000 random sets of desirable gambles. The error bars on the figures show

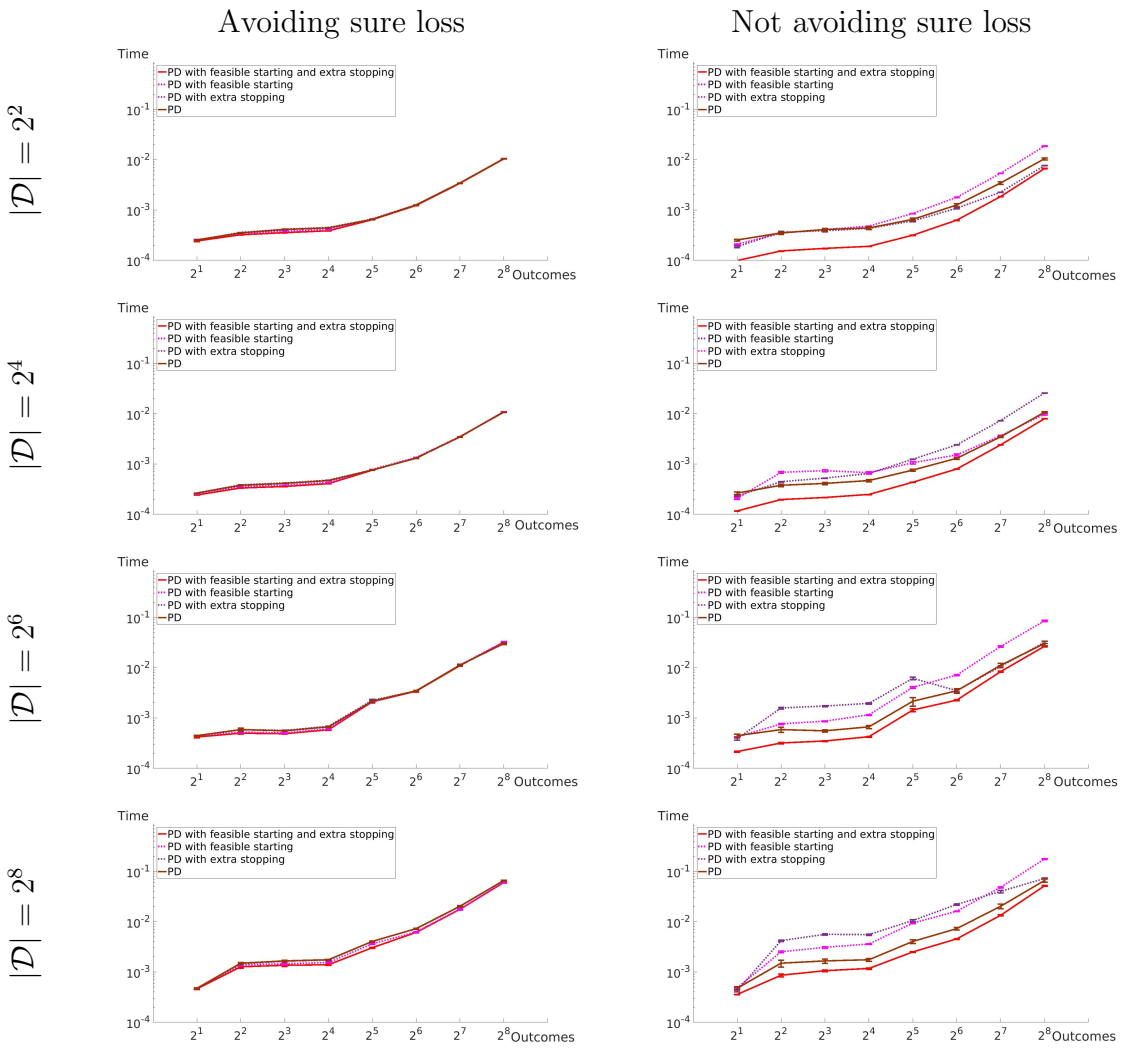

Figure 9.8: Comparison plots of the average computational time for different improved primal-dual methods (PD) for solving (P3) and (D5) where we fix the number of desirable gambles and vary the number of outcomes. The labels indicate different improved primal-dual methods.

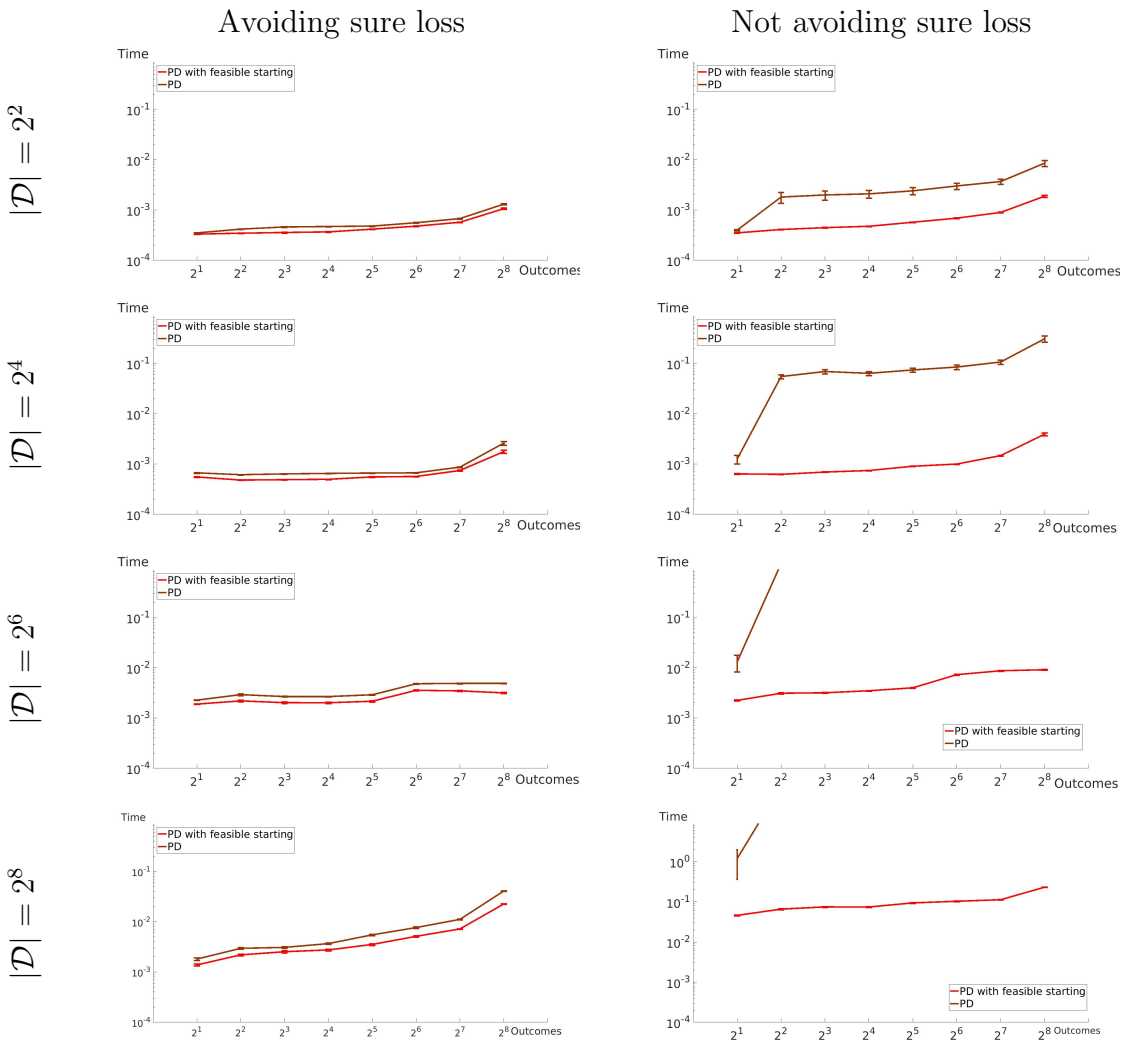

Figure 9.9: Comparison plots of the average computational time for an improved primal-dual method and a standard primal-dual method (PD) for solving (D4') and (P4') where we fix the number of desirable gambles and vary the number of outcomes. The labels indicate two different primal-dual methods.

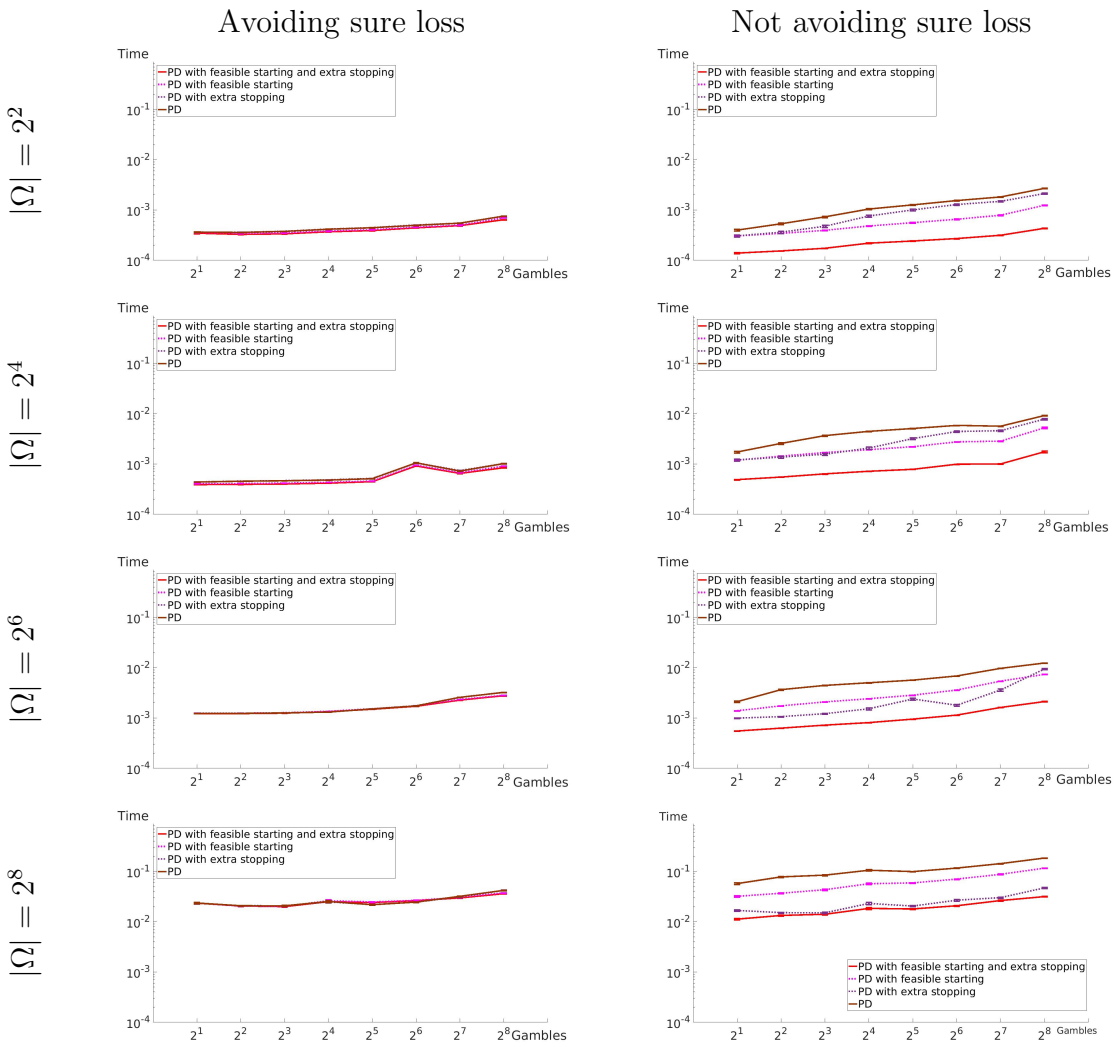

Figure 9.10: Comparison plots of the average computational time for different improved primal-dual methods (PD) for solving (P3) and (D5) where we fix the number of outcomes and vary the number of desirable gambles. The labels indicate different primal-dual methods.

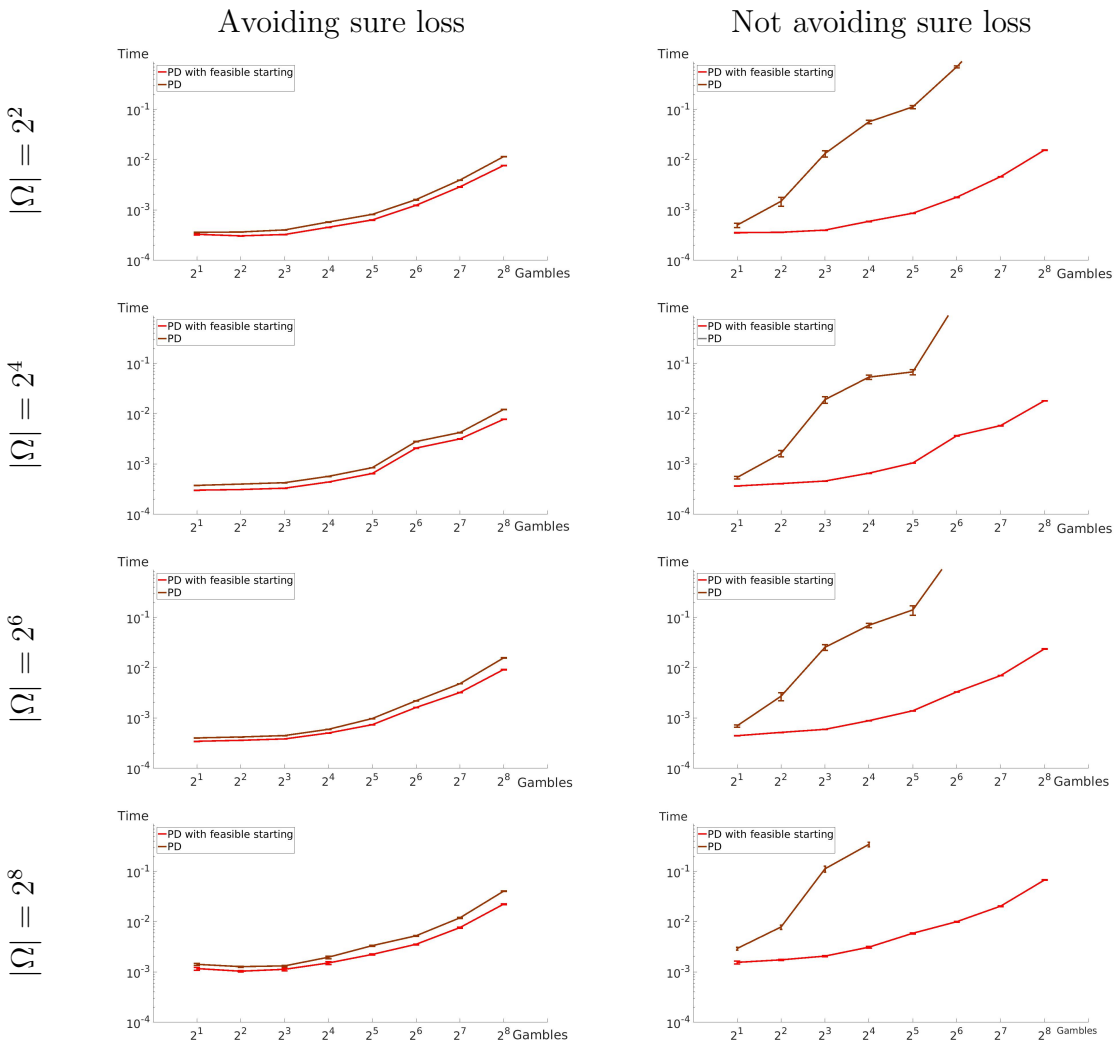

Figure 9.11: Comparison plots of the average computational time for an improved primal-dual method and a standard primal-dual method (PD) for solving (D4') and (P4') where we fix the number of outcomes and vary the number of desirable gambles. The labels indicate two different primal-dual methods.

approximate 95% confidence intervals on the mean computation time. These bars are barely visible as the sample size is large, except in few cases.

9.2 Discussion

The discussion in this section closely follows from the discussions in [26, 27]. According to our numerical results, the relative performance of the three methods depends on the number of desirable gambles and the number of outcomes. Specifically, if the number of outcomes is much larger than the number of desirable gambles, then solving either (D3) or (D4') is faster than solving (P3). However, if the number of outcomes is much less than the number of desirable gambles, then we prefer to solve (P3). From this result, we conclude that for checking avoiding sure loss, these algorithms solve linear programs which have the number of constraints less than the number of variables faster than solving linear programs where the number of variables is larger than the number of constraints. When the two numbers are of similar magnitude, there is no clear difference.

Even though the simplex method can easily find a basic feasible starting point, it cannot apply the extra stopping criterion. In the results, the simplex method is outperformed by the affine scaling and the primal-dual methods in most scenarios. Hence, the simplex method is not a good choice for checking avoiding sure loss.

On the other hand, the affine scaling and the primal-dual methods can benefit from these improvements. Specifically, when we solve (P3), these two methods can apply the extra stopping criterion and a simple way to obtain feasible starting points. In this case, the primal-dual method performs very well, especially when we do not avoid sure loss and the number of desirable gambles is large.

When we solve (D4'), these two interior-point methods can easily construct feasible starting points. In this case, the affine scaling method performs very well in

small problems whilst the primal-dual method does better when the problems are bigger.

Overall, if linear programs are small, then there is no big difference in the time taken to solve either (P3) or (D4'), and there is also no big difference between the performance of the methods. When the linear programs are large, the primal-dual method is the best choice. In this case, if the number of desirable gambles is large, then we solve (P3), and if the number of outcomes is large, then we solve (D4').

We discuss the impact of using our improvements in the primal-dual method as follows. When the sets avoid sure loss, using our feasible starting point in the primal-dual method shows a very slight improvement, regardless of which linear programs. Similarly, the extra stopping criterion does not help at all in this case, quite logically so, because it will never be invoked in this case. This implies that it also does not hinder performance; the overhead of the extra check is thus negligible.

Consider the case that sets do not avoid sure loss. When we solve (P3), both the extra stopping criterion and the feasible starting point considerably improve performance. Using both improvements gives the best results. Similarly, when we solve (D4'), the primal-dual method with feasible starting point performs much better than the standard primal-dual method. Therefore, the feasible starting point definitely improves its performance.

Note that these improvements benefit all applied problems as they reduce the computational burden of the original algorithms. Our benchmarking study quantified these benefits for a wide range of situations. In the case of not avoiding sure loss, we tested the hardest case where only a single gamble violates consistency. Any positive computational gain in these cases implies at least as large a gain for more general applied cases where multiple gambles violate consistency.

Also note that we can make the procedure 1. in algorithm 1 more general.

Specifically, if for $i = 1, \ldots, n$ we assign R_i a gamma $\mathrm{Ga}(a_i, 1)$ distribution, independently, and set $p(\omega_i) \coloneqq \frac{R_i}{\sum_{j=1}^{n} R_j}$, then this will give $p(\omega_1), \ldots, p(\omega_n)$ a Dirichlet $\mathrm{Dir}(a_1, a_2, \ldots, a_n)$ distribution [8, p.12]. This would allow us to change both the mean and the variance of $p(\omega_1), \ldots, p(\omega_n)$ and the resulting sets of desirable gambles might have different, collective, characteristics. This would affect the comparison of the algorithms in the benchmarking as some algorithms might tend to do better, or worse, in particular regions of the space of possible sets of desirable gambles. This might provide a means of further exploration of the comparison.

9.3 Summary

The first contribution based on [26, 27] is summarised as follows. We studied linear programming problems for checking avoiding sure loss. We then discussed and improved the simplex, the affine scaling and the primal-dual methods to efficiently solve linear programs for checking avoid sure loss. The improvements of these methods are, namely, (i) an extra stopping criterion, and (ii) a simple and quick algorithm for finding feasible starting points in these methods.

To measure our improvements, we also gave several algorithms for generating random sets of desirable gambles that either avoid or do not avoid sure loss. We compared the performance of different methods on generated sets.

We presented the relative performance of the three methods as a function of the number of desirable gambles and the number of outcomes. Overall, the affine scaling and primal-dual methods which benefit from the improvements outperform the simplex method in most cases. Therefore, the simplex method is not a good choice for solving linear programs for checking avoiding sure loss. If problems are small, then there is no big difference in performance between all methods. For large problems, our improved primal-dual method performs at least three times faster

than any of the other methods.

In the next contribution, we will look at checking avoiding sure loss for betting schemes, where we view betting odds and free coupons as a set of desirable gambles. For that specific problem, in addition to checking avoiding sure loss by solving a linear programming problem, we will show that we can calculate the natural extension for checking avoiding sure loss through the Choquet integral.

Contribution II

Chapter 10

Betting odds and free coupons

This chapter closely follows [29] which we publish in International Journal of Approximate Reasoning, volume 106, which is available in March 2019. In that work, we study whether and how customers can exploit betting odds and free coupons in order to make a sure gain. To start, in section 10.1 we introduce a betting scheme and explain fractional fixed odds. As betting odds can be viewed as a set of desirable gambles, we also check whether such a set avoids sure loss or not. Next, in section 10.2 we view a free coupon as part of a desirable gamble which is added to the set that avoids sure loss, and then we check whether such a larger set still avoids sure loss or not through the natural extension. We show that we can apply the Choquet integral to calculate natural extensions, and, with the complementary slackness conditions, to find a combination of bets in order to make a guaranteed gain. Finally, in section 10.3, we illustrate our results via some actual betting odds and free coupons in the market.

10.1 Betting schemes

This section closely follows [29, §3]. In the UK, a bookmaker usually offers fixed fractional odds on possible outcomes of an event that customers are interested in. For

example, in the European Football Championship 2016, customers are interested in the winner of the championship. Suppose that a bookmaker sets odds on England, say 17/2, and one customer accepts this odds. For every stake of £2 that the customer bets on England, he will win £17 plus the return of his stake. So the bookmaker will lose £17 in total when England is the winner of the championship. Otherwise, the bookmaker will pay nothing and keep £2.

Suppose that a customer accepts odds a/b on an outcome x, he can simply calculate his return as follows. For every amount b that the customer bets, he will either gain a plus the return of his stake (in case the bet is won, i.e. the true outcome is x) or get nothing (in case the bet is lost, i.e. the true outcome is not x). The bookmaker often writes $a/1$ as a. As the bookmaker accepts this transaction, the total payoff can be seen as a desirable gamble, say g, to the bookmaker:

$$g(\omega) = \begin{cases} -a & \text{if } \omega = x \\ b & \text{otherwise.} \end{cases} \tag{10.1}$$

Note that $-g$ can be viewed as the total payoff to the customer, so $-g$ is a desirable gamble to the customer.

Throughout this chapter, let $\Omega = \{\omega_1, \ldots, \omega_n\}$ be a finite set of outcomes. Suppose that for each i, the bookmaker sets a_i/b_i to be the betting odds on ω_i. By eq. (10.1), these odds can be viewed as a set of desirable gambles $\mathcal{D} = \{g_1, \ldots, g_n\}$, where

$$g_i(\omega) \coloneqq \begin{cases} -a_i & \text{if } \omega = \omega_i \\ b_i & \text{otherwise.} \end{cases} \tag{10.2}$$

Let a_i/b_i be the odds on ω_i. Suppose that we want to modify the denominator in this odds to be b_j. To do so, we can multiply a_i/b_i by b_j/b_j to be

$$a_i b_j / b_i b_j = \left(\frac{a_i b_j}{b_i} \right) / b_j. \tag{10.3}$$

Are the new odds still desirable? By the rationality axioms for desirability, the modified odds are still desirable; see lemma 47 below.

Lemma 47. [29, Lemma 2] Let a/b be desirable odds on an outcome $\tilde{\omega}$. Then, for all $\alpha > 0$, the odds $\alpha a/\alpha b$ on $\tilde{\omega}$ are also desirable. □

Proof. Consider the desirable gamble corresponding to the odds a/b:

$$g(\omega) \coloneqq \begin{cases} -a & \text{if } \omega = \tilde{\omega} \\ b & \text{otherwise.} \end{cases} \tag{10.4}$$

By rationality axiom (D3), for any $\alpha > 0$, the gamble αg is also desirable. Hence, the corresponding odds $\alpha a/\alpha b$ are also desirable. □

Lemma 47 will be very useful in the case that we want to modify odds to have the same denominator.

Before the bookmaker announcing betting odds for all possible outcomes in Ω, the bookmaker may want to check whether there is a combination of bets from which customers can make a sure gain. In other words, the bookmaker wants to check whether he avoids sure loss or not [46, Appendix 1, I4, p. 635]:

Theorem 48. [29, Theorem 5] Let $\Omega = \{\omega_1, \ldots, \omega_n\}$. Suppose a_i/b_i are betting odds on ω_i. For each $i \in \{1, \ldots, n\}$, let

$$g_i(\omega) \coloneqq \begin{cases} -a_i & \text{if } \omega = \omega_i \\ b_i & \text{otherwise,} \end{cases} \tag{10.5}$$

be the gamble corresponding to the odds a_i/b_i. Then $\mathcal{D} \coloneqq \{g_1, \ldots, g_n\}$ avoids sure loss if and only if

$$\sum_{i=1}^{n} \frac{b_i}{a_i + b_i} \geq 1. \tag{10.6}$$

□

Proof. Theorem 48 follows from theorem 51 (proved after) for $m = 1$. (Note that theorem 48 is not used in the proof of theorem 51.) □

Note that, at least $\sum_{i=1}^{n} \frac{b_i}{a_i+b_i}$ is equal to 1; otherwise bookmakers do not avoid sure loss. However, in practice, $\sum_{i=1}^{n} \frac{b_i}{a_i+b_i}$ is normally strictly greater than 1 and

$$100 \times \left(\sum_{i=1}^{n} \frac{b_i}{a_i + b_i} - 1 \right) \tag{10.7}$$

is called the *over-round margin* [6, 45].

Let us see an example of theorem 48.

Example 49. Suppose that a bookmaker provides betting odds 4/9 for W, 7/2 for D, and 8/1 for L. As

$$\frac{9}{4+9} + \frac{2}{7+2} + \frac{1}{8+1} = 1.026 \geq 1, \tag{10.8}$$

by theorem 48, the bookmaker avoids sure loss. Therefore, a customer cannot exploit these odds in order to make a sure gain. □

Note that the condition for avoiding sure loss of $\mathcal{D}$ in theorem 48, which is a set of desirable gambles derived from betting odds, is exactly the same as the condition for avoiding sure loss of $\underline{P}_{\overline{p}}$ in theorem 22. This condition is also equivalent to Proposition 4 in Cortis [4].

Next, we show that those odds can be modelled through an upper probability mass function:

Lemma 50. [29, Lemma 3] Let $\Omega = \{\omega_1, \ldots, \omega_n\}$, let $\omega_i \in \Omega$ and let g be the corresponding gamble to the odds on ω_i defined as in eq. (10.2), that is,

$$g_i(\omega) \coloneqq \begin{cases} -a_i & \text{if } \omega = \omega_i \\ b_i & \text{otherwise,} \end{cases} \tag{10.9}$$

where a_i and b_i are non-negative. If p is a probability mass function, that is $\sum_{\omega\in\Omega} p(\omega) = 1$ and $p(\omega) \geq 0$ for all $\omega \in \Omega$, then

$$\sum_{\omega\in\Omega} g_i(\omega)p(\omega) \geq 0 \qquad \iff \qquad \frac{b_i}{a_i + b_i} \geq p(\omega_i). \tag{10.10}$$

□

Proof. Suppose that $\sum_{\omega\in\Omega} p(\omega) = 1$ and $p(\omega_i) \geq 0$ for all i, then

$$\sum_{\omega\in\Omega} g_i(\omega)p(\omega) \geq 0 \iff -a_i p(\omega_i) + b_i \sum_{\omega\neq\omega_i} p(\omega) \geq 0 \tag{10.11}$$

$$\iff -a_i p(\omega_i) + b_i(1 - p(\omega_i)) \geq 0 \tag{10.12}$$

$$\iff \frac{b_i}{a_i + b_i} \geq p(\omega_i). \tag{10.13}$$

□

In order to avoid sure loss, the odds a_i/b_i on ω_i must satisfy eq. (10.13) [46, §3.3.3 (a)] (see the proof of theorem 51 for more detail). Therefore, the collection of these odds can be viewed as an upper probability mass function, that is,

$$\forall i \in \{1, \ldots, n\} \colon \overline{p}(\omega_i) \coloneqq \frac{b_i}{a_i + b_i}. \tag{10.14}$$

In general, there is more than one bookmaker in the market. The next challenge is whether a customer can exploit odds from different bookmakers in order to make a sure gain. To do so, we model betting odds from different bookmakers as a set of desirable gambles, and we check avoiding sure loss of this set. We recover the known result that it is optimal to pick maximal odds on each outcome [45]. Since greater odds are corresponding to a higher payoff to a customer, a sensible strategy for the customer is to choose the greatest odds on each outcome.

Theorem 51. [29, Theorem 6] Let $\Omega = \{\omega_1, \ldots, \omega_n\}$. Suppose that there are m different bookmakers. For each $k \in \{1, \ldots, m\}$, let a_{ik}/b_{ik} be the betting odds on

ω_i provided by bookmaker k. For each $i \in \{1, \ldots, n\}$ and $k \in \{1, \ldots, m\}$, let

$$g_{ik}(\omega) \coloneqq \begin{cases} -a_{ik} & \text{if } \omega = \omega_i \\ b_{ik} & \text{otherwise.} \end{cases} \tag{10.15}$$

be the desirable gamble corresponding to the odds a_{ik}/b_{ik}. Let a_i^*/b_i^* be the maximal betting odds on outcome ω_i, that is,

$$a_i^*/b_i^* \coloneqq \max_{k=1}^{m} \{a_{ik}/b_{ik}\}. \tag{10.16}$$

Then the set of desirable gambles $\mathcal{D} = \{g_{ik} \colon i \in \{1, \ldots, n\},\ k \in \{1, \ldots, m\}\}$ avoids sure loss if and only if

$$\sum_{i=1}^{n} \frac{b_i^*}{a_i^* + b_i^*} \geq 1. \tag{10.17}$$

□

Proof. See appendix B.4. □

Therefore, to check avoiding sure loss of several bookmakers, by theorem 51, we only need to consider the greatest odds on each outcome. Let us see an example.

Example 52. [29, Example 2] Suppose that in the market there are three bookmakers providing different odds for outcomes W, D, and L as in table 10.1.

Outcomes	Betting companies			Maximum odds
	River	Mountain	Forest	
W	4/5	17/20	3/4	17/20
D	13/5	14/5	13/5	14/5
L	10/3	3	16/5	10/3

Table 10.1: Table of odds provided by three bookmakers [29, Table 1].

Let $\mathcal{D}$ be the set of desirable gambles corresponding to all of these odds. Note that

the maximal betting odds are 17/20 for W, 14/5 for D and 10/3 for L. As

$$\frac{20}{17+20}+\frac{5}{14+5}+\frac{3}{10+3}=1.034\geq 1, \tag{10.18}$$

by theorem 51, we conclude that $\mathcal{D}$ avoids sure loss. Therefore, a customer cannot exploit these odds to make a sure gain. □

Suppose that there is a customer who is interested in odds provided by the three bookmakers as in table 10.1. A sensible strategy to him is to choose the greatest odds on each outcome. However, this implies that the customer will never pick any odds provided by Forest, because all of Forest's odds are less than or equal the odds provided by other bookmakers. Therefore, to encourage customers to bet with them, Forest may offer free coupons, which can be spent on betting, to the customer under certain conditions. In the next section, we will look at these free coupons in more detail.

10.2 Free coupons for betting

A free coupon is a free stake that is given by a bookmaker to a customer under some condition. For example, a free coupon is offered to a customer who first bets with a bookmaker. The free coupon can be spent on some betting odds that the customer wants to bet. In fact, the free coupon is not truly free, since the customer firstly has to bet on some odds before he claims the free coupon. In addition, the bookmakers usually set some requirements, for instance, a limit on the amount of free coupons that customers can claim, or a restriction of choices that customers can spend their free coupons.

In this study, we are interested in whether customers can exploit those given odds and free coupons in order to find a strategy of betting that incurs a sure gain. If there is a possible way to do that, then we will find an algorithm that gives such

a strategy. Note that the rest of this section closely follows [29, §4].

For simplicity in this study, we decide standard requirements for claiming free coupons from the bookmakers as follows:

1. Once the customer has placed his first bet, the bookmaker will give him a free coupon whose value is equal to the value of the bet that he placed.
2. The bookmaker announces the maximum value of the free coupon.
3. The free coupon only applies to the customer's first bet with the bookmaker.
4. The customer must spend his free coupon with the same bookmaker on other outcomes.
5. The customer must spend his free coupon on only a single outcome.

Here is an example of claiming free coupons.

Example 53. (Adapted from [29, Example 3]) Suppose that Forest has the following offer: a free coupon will be given to a customer who first bets with Forest, and the value of the coupon is equal to the value of the first bet that the customer placed. The maximum value of the free coupon is £100 and the customer must spend his free coupon on only a single outcome.

From table 10.1, suppose that Tim, who is a customer, has never bet with Forest and he decides to place £5 on the odds 13/5 of the outcome D, then he will pay £5 to Forest and he will claim a free coupon valued £5. Tim can use his free coupon to bet on other outcomes with Forest, that is either W or L, but he cannot separate his free coupon to bet on both of them. □

Once Tim receives a free coupon, he can spend his free coupons as in the next example.

Example 54. (Adapted from [29, Example 4]) Continuing from the previous example, Tim has his free coupon valued £5 from Forest. Since Tim must spend his

free coupon valued £5 on only a single outcome, by lemma 47, we modify odds 3/4 by multiplying them by 5/5. Now all odds have the same denominator which is 5.

Outcomes	W	D	L
odds	$(\frac{3 \cdot 5}{4})/5$	13/5	16/5

Table 10.2: Table of modified odds [29, Table 2].

If Tim spends his free coupon to bet on L and the true outcome is L, then Forest will lose £16; otherwise Forest will lose nothing. On the other hand, if Tim spends the coupon to bet on W and the true outcome is W, then Forest will lose £$\frac{3 \cdot 5}{4}$; otherwise Forest will lose nothing. The total payoff to Forest is summarised in table 10.3.

Betting a free coupon on	Outcomes		
	W	D	L
L	0	0	-16
W	$-\frac{3 \cdot 5}{4}$	0	0

Table 10.3: Table of total payoff to Forest [29, Table 3].

□

Suppose that the customer first bets £b_i on an outcome ω_i with corresponding odds a_i/b_i. The payoff to the bookmaker is represented as a gamble g_{ω_i} in the table 10.4. As this is his first bet, the customer obtains a free coupon valued b_i, and he can spend this free coupon to bet on a single outcome. Suppose that he bets on ω_j with corresponding odds a_j/b_j. As the denominators are not necessarily equal, we multiply odds a_j/b_j by $\frac{b_i}{b_i}$. The modified odds are $(\frac{a_j \cdot b_i}{b_j})/b_i$. Note that as the free coupon must be spent on other outcomes, ω_j cannot coincide with ω_i.

If the true outcome is ω_j, then the bookmaker will lose $\frac{a_j \cdot b_i}{b_j}$. Otherwise the bookmaker will gain nothing. This payoff corresponding to the free coupon to the

bookmaker is viewed as a gamble $\tilde{g}_{\omega_j}$ in the table 10.4. As g_{ω_i} and $\tilde{g}_{\omega_j}$ are desirable to the bookmaker, by rationality axiom (D4), $g_{\omega_i} + \tilde{g}_{\omega_j}$ is also desirable.

Outcomes	ω_i	ω_j	others
g_{ω_i}	$-a_i$	b_i	b_i
$\tilde{g}_{\omega_j}$	0	$-\frac{a_j \cdot b_i}{b_j}$	0
$g_{\omega_i} + \tilde{g}_{\omega_j}$	$-a_i$	$\frac{(b_j - a_j)b_i}{b_j}$	b_i

Table 10.4: Table of the first-free desirable gamble to the bookmaker [29, Table 4].

We denote $g_{\omega_i\omega_j} \coloneqq g_{\omega_i} + \tilde{g}_{\omega_j}$ and call it the *first-free* desirable gamble to the bookmaker. Note that $-g_{\omega_i\omega_j}$ is desirable to the customer. Also note that the customer can still bet on other odds, but he will not gain any free coupon from his additional bets. This is because the bookmaker gives him the free coupon only once.

In the actual market, there is usually more than one bookmaker offering a free coupon. Therefore, the customer can first bet with different bookmaker in order to gain several free coupons. These can be viewed as a total first-free desirable gamble combining from several first-free desirable gambles. In this study, we only consider the case that the customer first bets and claims a free coupon from a single bookmaker. In this case, we face a combinatorial problem over all first-free desirable gambles.

We want to check whether the bookmaker avoids sure loss or not for this situation. In other words, we check whether $\mathcal{D} \cup \{g_{\omega_i\omega_j}\}$ avoids sure loss or not. Recall that theorem 30 implies that: if $\mathcal{D}$ avoids sure loss, then $\mathcal{D} \cup \{g_{\omega_i\omega_j}\}$ avoids sure loss if and only if $\overline{E}(g_{\omega_i\omega_j}) \geq 0$. In the case that $\mathcal{D} \cup \{g_{\omega_i\omega_j}\}$ does not avoid sure loss, by theorem 30, the bookmaker will lose at least $|\overline{E}(g_{\omega_i\omega_j})|$ which is the customer's highest sure gain. Therefore, the customer can combine $g_{\omega_i\omega_j}$ and a combination of g_i to yield a sure gain $|\overline{E}(g_{\omega_i\omega_j})|$ from the bookmaker.

Let f be any first-free desirable gamble to the bookmaker. To apply theorem 30, we have to check whether $\mathcal{D}$ avoids sure loss. If $\underline{P}_{\overline{p}}$ does not avoid sure loss, then without a free coupon, there is a non-negative combination of gambles that the customer can exploit to make a sure gain. On the other hand, if $\underline{P}_{\overline{p}}$ avoids sure loss, then we can write f in terms of its level sets and use corollary 28 to calculate the natural extension of f.

Example 55. (Adapted from [29, Example 5]) Recall that Forest provides betting odds on W, D, and L as in table 10.1. By eq. (10.14), we have

$$\overline{p}(W) = \frac{4}{7} \qquad \overline{p}(D) = \frac{5}{18} \qquad \overline{p}(L) = \frac{5}{21}. \tag{10.19}$$

Since $\overline{p}(W) + \overline{p}(D) + \overline{p}(L) \geq 1$, $\underline{P}_{\overline{p}}$ avoids sure loss by theorem 22 which coincides with example 52.

Continuing from example 54, suppose that Tim first bets on D and spends his free coupon to bet on L. Then, the first-free desirable gamble g_{DL} to Forest is as follows:

Outcomes	W	D	L
g_D	5	-13	5
g_L	0	0	-16
g_{DL}	5	-13	-11

Table 10.5: Table of desirable gambles to Forest [29, Table 5].

We decompose g_{DL} in terms of its level sets as

$$g_{DL} = -13I_{A_0} + 2I_{A_1} + 16I_{A_2} \tag{10.20}$$

where $A_0 = \{W, D, L\}$, $A_1 = \{W, L\}$ and $A_2 = \{W\}$. By theorem 26, we have

$$\overline{E}_{\overline{p}}(A_0) = \min\{\overline{p}(W) + \overline{p}(D) + \overline{p}(L), 1\} = 1 \tag{10.21}$$

$$\overline{E}_{\overline{p}}(A_1) = \min\{\overline{p}(W) + \overline{p}(L), 1\} = \frac{17}{21} \tag{10.22}$$

$$\overline{E}_{\overline{p}}(A_2) = \min\{\overline{p}(W), 1\} = \frac{4}{7}. \tag{10.23}$$

Substitute $\overline{E}_{\overline{p}}(A_i)$, $i \in \{0, 1, 2\}$ into eq. (10.20). By corollary 28, we have

$$\overline{E}_{\overline{p}}(g_{DL}) = -13\overline{E}_{\overline{p}}(A_0) + 2\overline{E}_{\overline{p}}(A_1) + 16\overline{E}_{\overline{p}}(A_2) = -\frac{47}{21}. \tag{10.24}$$

As $\overline{E}_{\overline{p}}(g_{DL}) = -\dfrac{47}{21} < 0$, by theorem 30, Forest does not avoid sure loss. Therefore, spending his free coupon, there is a combination of bets for Tim to make a sure gain. □

Next, we explain a strategy for Tim to bet in order to get $£\dfrac{47}{21}$. Remember that $\Omega = \{\omega_1, \ldots, \omega_n\}$ and that g_i is the corresponding gamble to the odds a_i/b_i on ω_i:

$$g_i(\omega) = \begin{cases} -a_i & \text{if } \omega = \omega_i \\ b_i & \text{otherwise.} \end{cases} \tag{10.25}$$

Also recall that we can calculate $\overline{E}_{\overline{p}}(f)$, or $\overline{E}_{\mathcal{D}_{\underline{P}_{\overline{p}}}}(f)$, by definition 25, for any gamble f by solving the following linear program:

$$\text{(S)} \qquad \min \quad \alpha \tag{Sa}$$

$$\text{subject to} \quad \begin{cases} \forall \omega \in \Omega \colon \alpha - \sum_{i=1}^{n} g_i(\omega)\lambda_i \geq f(\omega) \\ \forall i = 1, \ldots, n \colon \lambda_i \geq 0, \end{cases} \tag{Sb}$$

where the optimal value of α gives $\overline{E}_{\overline{p}}(f)$. If the optimal value of α is strictly negative, then the optimal values of $\lambda_1, \ldots, \lambda_n$ give a combination of bets for a

customer to make a sure gain. The dual of (S) is given:

$$\text{(T)} \qquad \max \quad \sum_{\omega\in\Omega} f(\omega)p(\omega) \tag{Ta}$$

$$\text{subject to} \quad \begin{cases} \forall g_i\colon \sum_{\omega\in\Omega} g_i(\omega)p(\omega) \geq 0 \\ \sum_{\omega\in\Omega} p(\omega) = 1 \\ \forall \omega\colon p(\omega) \geq 0. \end{cases} \tag{Tb1}$$

We apply lemma 50 to the constraints in eq. (Tb1) to be:

$$\text{subject to} \quad \begin{cases} \forall \omega\colon 0 \leq p(\omega) \leq \overline{p}(\omega) \\ \sum_{\omega\in\Omega} p(\omega) = 1. \end{cases} \tag{Tb2}$$

We see that the objective function eq. (Ta) is $E_p(f)$ which is the expectation of f with respect to the probability mass function p. As the optimal value of (T) is $\overline{E}_{\overline{p}}(f)$, if we can find a p that satisfies (T)'s constraints eq. (Tb2) and $\overline{E}_{\overline{p}}(f) = E_p(f)$, then we have found an optimal solution of (T).

In algorithm 7, we first construct a p, by assigning as much mass as possible to the smallest level sets. Then, in theorem 56, we show that this p satisfies eq. (Tb2) and $\overline{E}_{\overline{p}}(f) = E_p(f)$.

We then show that p in eq. (10.30) satisfies eq. (Tb2) and $\overline{E}_{\overline{p}}(f) = E_p(f)$.

Theorem 56. [29, Theorem 7] The probability mass function p defined by eq. (10.30) satisfies eq. (Tb2) and $\overline{E}_{\overline{p}}(f) = E_p(f)$. □

Proof. Let $\Omega = \{\omega_1, \ldots, \omega_n\}$ be ordered as in eq. (10.27), and let k be the smallest index such that $\sum_{j=1}^{k} \overline{p}(\omega_j) \geq 1$. By eq. (10.30), $\sum_{i=1}^{n} p(\omega_i) = 1$ and

$$p(\omega_k) = 1 - \sum_{j=1}^{k-1} \overline{p}(\omega_j) \leq \sum_{j=1}^{k} \overline{p}(\omega_j) - \sum_{j=1}^{k-1} \overline{p}(\omega_j) = \overline{p}(\omega_k), \tag{10.31}$$

Algorithm 7 Construct an optimal solution p of (T) [29, Algorithm 1]

Input: A gamble f, a set of outcomes Ω.
Output: An optimal solution p of (T).

1. Rewrite f as
$$f = \sum_{i=0}^{m} \lambda_i A_i \tag{10.26}$$
where $\Omega = A_0 \supsetneq A_1 \supsetneq \cdots \supsetneq A_m \supsetneq \emptyset$ are the level sets of f and $\lambda_0 \in \mathbb{R}$, $\lambda_1, \ldots, \lambda_m > 0$.
2. Order $\omega_1, \omega_2, \ldots, \omega_n$ such that
$$\forall i \leq j \colon A_{\omega_i} \subseteq A_{\omega_j}, \tag{10.27}$$
where A_ω is the smallest level set to which ω belongs, that is
$$A_\omega = \bigcap_{\substack{i=0 \\ \omega \in A_i}}^{m} A_i. \tag{10.28}$$
So, we start with those ω in A_m, then those in $A_{m-1} \setminus A_m$, then those in $A_{m-2} \setminus A_{m-1}$, and so on.
3. Let k be the smallest index such that
$$\sum_{j=1}^{k} \overline{p}(\omega_j) \geq 1. \tag{10.29}$$
There is always such k because $\underline{P}_{\overline{p}}$ avoids sure loss. Define p as follows:
$$p(\omega_i) \coloneqq \begin{cases} \overline{p}(\omega_i) & \text{if } i < k \\ 1 - \sum_{j=1}^{i-1} \overline{p}(\omega_j) & \text{if } i = k \\ 0 & \text{if } i > k. \end{cases} \tag{10.30}$$

so for all $i \in \{1, \ldots, n\}$, $0 \leq p(\omega_i) \leq \overline{p}(\omega_i)$. Therefore, p satisfies eq. (Tb2). Next, we will show that for all level sets A_i,

$$\min \left\{ \sum_{\omega \in A_i} \overline{p}(\omega), 1 \right\} = E_p(A_i). \tag{10.32}$$

Remember that A_{ω_k} is the smallest level set that contains ω_k. By eq. (10.30), for all $A_i \subsetneq A_{\omega_k}$, we know that $p(\omega) = \overline{p}(\omega)$ for all $\omega \in A_i$, and so

$$\min \left\{ \sum_{\omega \in A_i} \overline{p}(\omega), 1 \right\} = \sum_{\omega \in A_i} \overline{p}(\omega) = \sum_{\omega \in A_i} p(\omega). \tag{10.33}$$

For all $A_i \supseteq A_{\omega_k}$, we know that $\sum_{\omega \in A_i} p(\omega) = 1$ and $\sum_{\omega \in A_i} \overline{p}(\omega) \geq 1$, so

$$\min \left\{ \sum_{\omega \in A_i} \overline{p}(\omega), 1 \right\} = 1 = \sum_{\omega \in A_i} p(\omega). \tag{10.34}$$

Hence, eq. (10.32) holds. Therefore,

$$\begin{aligned}
\overline{E}_{\overline{p}}(f) &= \sum_{i=0}^{m} \lambda_i \overline{E}(A_i) && \text{(by eq. (6.16))} && (10.35)\\
&= \sum_{i=0}^{m} \lambda_i \min \left\{ \sum_{\omega \in A_i} \overline{p}(\omega), 1 \right\} && \text{(by eq. (6.13))} && (10.36)\\
&= \sum_{i=0}^{m} \lambda_i E_p(A_i) && \text{(by eq. (10.32))} && (10.37)\\
&= E_p(f). && && (10.38)
\end{aligned}$$

□

To sum up, we can use eq. (10.30) to construct an optimal solution p of (T).

In general, the complementary slackness condition in theorem 13 can be used to find an optimal solution of the dual of (T) [50, p. 329]. Note that, as the dual problem (T) has an optimal solution and the dual problem is bounded, then by the strong duality theorem [33, p. 71], an optimal solution of the primal problem (S)

exists and achieves the same optimal value. In addition, a pair of solutions to (S) and (T) is optimal if, and only if, they satisfy the complementary slackness condition [7, p. 62]. Specifically, in our case, the condition holds for any non-negative variable and its corresponding dual constraint [10, p. 184, ll. 3–5]. In particular, precisely, let $p(\omega_1), \ldots, p(\omega_n)$ be any feasible solution of (T), and let α, $\lambda_1, \ldots, \lambda_n$ be any feasible solution of (S). Then, by complementary slackness, these solutions are optimal if, and only if, for all $j \in \{1, \ldots, n\}$, we have that

$$\left(\alpha - \sum_{i=1}^{n} g_i(\omega_j)\lambda_i - f(\omega_j)\right) p(\omega_j) = 0 \quad \text{and} \quad (\overline{p}(\omega_j) - p(\omega_j))\lambda_j = 0. \tag{10.39}$$

This is equivalent to

1. if $p(\omega_j) > 0$, then $\alpha - \sum_{i=1}^{n} g_i(\omega_j)\lambda_i = f(\omega_j)$, and
2. if $p(\omega_j) < \overline{p}(\omega_j)$, then $\lambda_j = 0$.

Therefore, if we have an optimal solution $p(\omega_1), \ldots, p(\omega_n)$ of (T) and the optimal value α, then we can use these equations as a system of equalities in $\lambda_1, \ldots, \lambda_n$. Note that some solutions of this system may not satisfy feasibility, i.e. they may violate $\lambda_i \geq 0$. However, all solutions of this system that satisfy $\lambda_i \geq 0$ are guaranteed to be optimal solutions of (S) [29, p. 139].

What does this system of equalities look like? Remember that in algorithm 7, k is defined as the smallest index such that $\sum_{j=1}^{k} \overline{p}(\omega_j) \geq 1$. According to eq. (10.30), for all $j \in \{1, \ldots, k-1\}$, we have that $p(\omega_j) > 0$, so we have the following equalities: for all $j \in \{1, \ldots, k-1\}$,

$$\alpha - \sum_{i=1}^{n} g_i(\omega_j)\lambda_i = f(\omega_j). \tag{10.40}$$

For all $j \in \{k+1, \ldots, n\}$ we have that $p(\omega_j) = 0 < \overline{p}(\omega_j)$, so $\lambda_j = 0$ for all $j \in \{k+1, \ldots, n\}$. For $j = k$, if $p(\omega_k) < \overline{p}(\omega_k)$, then we can also set $\lambda_k = 0$.

Otherwise, we know that $p(\omega_k) = \overline{p}(\omega_k) > 0$ and so we can simply impose the same equality as for $j \in \{1, \ldots, k-1\}$. In conclusion, let k' be the largest index j for which $p(\omega_j) = \overline{p}(\omega_j)$. Then as the optimal solution of (S) exists, it can be obtained by solving the following system:

$$\forall j \in \{1, \ldots, k'\} \colon \alpha - \sum_{i=1}^{k'} g_i(\omega_j)\lambda_i = f(\omega_j) \tag{10.41}$$

$$\forall j \in \{k'+1, \ldots, n\} \colon \lambda_j = 0. \tag{10.42}$$

So, effectively, all we are left with is a system of k' variables in k' constraints [29, p. 139].

Note that by lemma 47, we can modify the odds to have the same denominator (all b_i are equal), so it will be much easier to solve the new system.

Also note that in the first-free coupon scenario, to make a sure gain, the customer has to bet on every outcome. This implies that the only coefficients λ_i whose value can be zero are those corresponding to the gambles in the first-free gamble chosen by the customer. Hence, in that specific case, $k' \geq n - 2$ [29, p. 139].

Now, we go back to find a strategy for Tim to bet with Forest.

Example 57. [29, Example 6] Continuing from example 55, to calculate $\overline{E}(g_{DL})$, we can solve the following corresponding linear programs:

$$\text{(S1)} \qquad \min \quad \alpha \tag{S1a}$$

$$\text{subject to} \quad \begin{cases} \alpha + 3\lambda_W - 5\lambda_D - 5\lambda_L \geq 5 \\ \alpha - 4\lambda_W + 13\lambda_D - 5\lambda_L \geq -13 \\ \alpha - 4\lambda_W - 5\lambda_D + 16\lambda_L \geq -11 \end{cases} \tag{S1b}$$

$$\text{and} \quad \lambda_W \geq 0,\ \lambda_D \geq 0,\ \lambda_L \geq 0,\ \alpha \text{ free}, \tag{S1c}$$

$$\text{(T1)} \qquad \max \quad 5p(W) - 13p(D) - 11p(L) \tag{T1a}$$

$$\text{subject to} \quad \begin{cases} 0 \leq p(W) \leq 4/7 \\ 0 \leq p(D) \leq 5/18 \\ 0 \leq p(L) \leq 5/21 \\ p(W) + p(D) + p(L) = 1. \end{cases} \tag{T1b}$$

By eq. (10.27), we see that

$$A_W \subseteq A_L \subseteq A_D, \tag{10.43}$$

so an optimal solution of (T1) is obtained as follows:

$$p(W) = \frac{4}{7}, \quad p(L) = \frac{5}{21}, \quad p(D) = 1 - \left(\frac{4}{7} + \frac{5}{21}\right) = \frac{4}{21}. \tag{10.44}$$

As $p(W) = \overline{p}(W)$ and $p(L) = \overline{p}(L)$, whilst $p(D) < \overline{p}(D)$, by the complementary slackness, the optimal solution of (S1) must have $\lambda_D = 0$ and can be obtained by solving the following linear system:

$$\alpha + 3\lambda_W - 5\lambda_L = 5 \tag{P1b1}$$

$$\alpha - 4\lambda_W + 16\lambda_L = -11, \tag{P1b2}$$

where the value of α is $-\frac{47}{21}$. We solve this system and get an optimal solution: $\lambda_W = \frac{18}{7}$ and $\lambda_L = \frac{2}{21}$.

We can read a strategy for Tim to make a guaranteed gain from the optimal solution of (S1) as follows. Once he first bets £5 on D and claims a free coupon valued £5 to bet on L, he then additionally bets £$\frac{18}{7}$ on W and £$\frac{2}{21}$ on D. By this strategy, he definitely makes a sure gain of £$\frac{47}{21}$ from Forest regardless of the true outcome. □

So far, we have seen theoretical results with some examples of checking avoiding sure loss for betting odds and free coupons. Next, we will apply these theoretical results to some actual odds in the market. Indeed, we will check whether and how

a customer can exploit those actual odds and free coupons in order to make a sure gain. Note that the next section closely follows [29, §5]

10.3 Actual football betting odds

In 2016, the 15th UEFA European Championship was held in France from 10 June to 10 July 2016 where 24 teams participated. In this event, customers were interested in which team will be the winner of the European Championship.

Consider table C.1 which is in appendix C.1 where we list actual betting odds provided by 27 bookmakers on the winner of the European Football Championship 2016. From table C.1, the greatest betting odds on each outcome are listed in table 10.6.

Nations	Odds	Nations	Odds	Nations	Odds
France	10/3	Austria	45	Czech Republic	135
Germany	23/5	Poland	50	Slovakia	150
Spain	5	Switzerland	66	Rep of Ireland	170
England	9	Russia	85	Iceland	180
Belgium	57/5	Turkey	94	Romania	275
Italy	91/5	Wales	100	N Ireland	400
Portugal	20	Ukraine	100	Hungary	566
Croatia	27	Sweden	104	Albania	531

Table 10.6: Table of maximum betting odds for the European Football Championship 2016 [29, Table 6].

For all $i \in \{1, \ldots, 24\}$, let a_i^*/b_i^* be the maximal betting odds in table 10.6. As $\sum_{i=1}^{24} \frac{b_i^*}{a_i^*+b_i^*} = 1.0349 \geq 1$, by theorem 51, the set of desirable gambles derived from

the odds in table C.1 avoids sure loss. Therefore, there is no combination of bets which results in a sure gain. In this case, a customer cannot exploit odds from different bookmakers to make a sure gain.

Suppose that James is a new customer who is interested in betting with one of bookmakers, say Bet2 in table C.1. Since he has never bet with Bet2 before, Bet2 will give him a free coupon on his first bet with them. With the free coupon, we will find whether and how James can bet in order to make a guaranteed gain from Bet2.

Let $\mathcal{D}$ be a set of desirable gambles corresponding to the odds provided by Bet2. Let g be any first-free desirable gamble to Bet2. We will check whether $\mathcal{D} \cup \{g\}$ avoids sure loss or not. As there are 24 possible outcomes of this event, the total number of different first-free desirable gambles with Bet2 is $24 \times 23 = 552$.

Suppose that James first bets £1 on France and then spends his free coupon valued £1 on Spain. So, the first-free desirable gamble g_{FG} is

Outcomes	France	Spain	others
g_F	-3	1	1
$\tilde{g}_S$	0	-5	0
g_{FS}	-3	-4	1

Table 10.7: Table of James' first-free gamble [29, Table 7].

where F and S denote France and Spain respectively. Again, to calculate $\overline{E}(g_{FS})$ by the Choquet integral, we first decompose g_{FS} in terms of its level sets as

$$g_{FS} = -4I_{A_0} + I_{A_1} + 4I_{A_2} \tag{10.45}$$

where $A_0 = \Omega$, $A_1 = \Omega \setminus \{S\}$ and $A_2 = \Omega \setminus \{F, S\}$. By theorem 26, we have

$$\overline{E}(A_0) = 1 \qquad \overline{E}(A_1) = 0.9810 \qquad \overline{E}(A_2) = 0.7310. \tag{10.46}$$

By corollary 28, we substitute $\overline{E}(A_i)$, $i \in \{0, 1, 2\}$ to eq. (10.45) and obtain

$$\overline{E}(g_{FS}) = -4\overline{E}(A_0) + \overline{E}(A_1) + 4\overline{E}(A_2) = -0.0950. \tag{10.47}$$

Therefore, $\mathcal{D} \cup \{g_{FS}\}$ does not avoid sure loss.

Among all possible first-free gambles, we observe that there are three further gambles whose $\overline{E}$ is also less than zero, namely $\overline{E}(g_{FG}) = -0.2093$, $\overline{E}(g_{GF}) = -0.0117$ and $\overline{E}(g_{GS}) = -0.0950$, where G denotes Germany. So, by theorem 30, $\mathcal{D} \cup \{g\}$ does not avoid sure loss when $g \in \{g_{FS}, g_{FG}, g_{GF}, g_{GS}\}$; otherwise $\mathcal{D} \cup \{g\}$ avoids sure loss. Therefore, if

1. James first bets on France and then spends his free coupon to bet on either Spain or Germany, or

2. James first bets on Germany and then spends his free coupon to bet on either France or Spain,

then there is a combination of bets for him to bet in order to make a guaranteed gain from Bet2.

Consider the case where James first bets £1 on France and claims his free coupon valued £1 to bet on Spain. We can find a strategy for James to bet in order to make a sure gain £0.0950 as follows: We first construct an optimal solution of the corresponding problem (T) (see column $p(\omega_i)$ in table 10.8) through algorithm 7. Next, we can find the optimal solution of the corresponding problem (S) by using the optimal solution of (T) with the complementary slackness condition. After solving the linear system, the optimal solution of (S) is presented in column λ_i in table 10.8. Therefore, if James additionally bets on other teams as in column λ_i, then he will make a sure gain of £0.095 from Bet2.

Order ω_i	Nations	Odds	$\overline{p}(\omega_i)$	Optimal solutions	
				$p(\omega_i)$	λ_i
1	Germany	4	$\frac{1}{5}$	$\frac{1}{5}$	1
2	England	9	$\frac{1}{10}$	$\frac{1}{10}$	0.5
3	Belgium	10	$\frac{1}{11}$	$\frac{1}{11}$	$\frac{5}{11}$
4	Italy	16	$\frac{1}{17}$	$\frac{1}{17}$	$\frac{5}{17}$
5	Portugal	18	$\frac{1}{19}$	$\frac{1}{19}$	$\frac{5}{19}$
6	Croatia	25	$\frac{1}{26}$	$\frac{1}{26}$	$\frac{5}{26}$
7	Austria	40	$\frac{1}{41}$	$\frac{1}{41}$	$\frac{5}{41}$
8	Poland	50	$\frac{1}{51}$	$\frac{1}{51}$	$\frac{5}{51}$
9	Switzerland	40	$\frac{1}{41}$	$\frac{1}{41}$	$\frac{5}{41}$
10	Russia	66	$\frac{1}{67}$	$\frac{1}{67}$	$\frac{5}{67}$
11	Turkey	80	$\frac{1}{81}$	$\frac{1}{81}$	$\frac{5}{81}$
12	Wales	80	$\frac{1}{81}$	$\frac{1}{81}$	$\frac{5}{81}$
13	Ukraine	66	$\frac{1}{67}$	$\frac{1}{67}$	$\frac{5}{67}$
14	Sweden	80	$\frac{1}{81}$	$\frac{1}{81}$	$\frac{5}{81}$
15	Czech Republic	100	$\frac{1}{101}$	$\frac{1}{101}$	$\frac{5}{101}$
16	Slovakia	100	$\frac{1}{101}$	$\frac{1}{101}$	$\frac{5}{101}$
17	Rep of Ireland	150	$\frac{1}{151}$	$\frac{1}{151}$	$\frac{5}{151}$
18	Iceland	150	$\frac{1}{151}$	$\frac{1}{151}$	$\frac{5}{151}$
19	Romania	100	$\frac{1}{101}$	$\frac{1}{101}$	$\frac{5}{101}$
20	N Ireland	250	$\frac{1}{251}$	$\frac{1}{251}$	$\frac{5}{251}$
21	Albania	250	$\frac{1}{251}$	$\frac{1}{251}$	$\frac{5}{251}$
22	Hungary	250	$\frac{1}{251}$	$\frac{5}{251}$	$\frac{5}{251}$
23	France	3	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$
24	Spain	5	$\frac{1}{6}$	$\frac{586}{579}$	0

Table 10.8: Table of a summary of odds provided by Bet2, the upper probability mass function $\overline{p}(\omega_i)$, and optimal solutions of (T) and (S) [29, Table 8].

10.4 Summary

For this contribution, we studied whether and how a customer can exploit given betting odds and free coupons in order to make a sure gain from bookmakers. To do so, we viewed these odds and free coupons as a set of desirable gambles and applied theorem 30 to check whether such a set avoids sure loss or not via the natural extension. For this specific problem, we showed that we can easily calculate the natural extension through the Choquet integral by using corollary 28.

If the set does not avoid sure loss, then there is a combination of bets which results in a sure gain to customers. This combination can be derived from the optimal solution of the corresponding linear programming problem. In this case, we presented how to use the Choquet integral and the complementary slackness condition to directly obtain the desired combination of bets, without actually solving linear programming problems, but instead just solving a linear system of equalities. This technique can be applied to arbitrary problems involving upper probability mass functions.

We applied the results to some actual betting odds in the market, that is, betting odds on the winning of the European Football Championship 2016. We found that any sets of desirable gambles derived from those odds avoid sure loss. However, with a free coupon, we identified sets of desirable gambles that no longer avoid sure loss. Consequently, in this case, when a free coupon is added, there was a combination of bets from which the customer could have made a sure gain.

Contribution III

Chapter 11

Improving algorithms for finding maximal gambles

This chapter closely follows [28, §3] which we submitted to International Journal of Approximate Reasoning on 10 December 2018. The aim of this chapter is first to study several algorithms from the literature for finding maximal gambles and propose a new algorithm for finding maximal gambles in section 11.1. Next, in section 11.2 we study an algorithm for finding interval dominant gambles which can be used to eliminate non-maximal gambles. Finally, to speed up the process of evaluating natural extensions in these algorithms, we investigate the structure of linear programs and propose an improvement in section 11.3.

11.1 Algorithms for finding maximal gambles

In this section, we discuss algorithms for finding $\text{opt}_{\succ}(\mathcal{K})$. We first study two algorithms from the literature, and then propose a new algorithm based on a suggestion from Troffaes and Hable [40, p. 336].

One can see that a gamble f is maximal in $\mathcal{K}$ only if

$$\forall g \in \mathcal{K} : \overline{E}(f - g) \geq 0. \tag{11.1}$$

Suppose that there are m possible outcomes in Ω, k gambles in $\mathcal{K}$ and n gambles in $\operatorname{dom} \underline{P}$ where $\underline{P}$ avoids sure loss. In order to determine whether f is maximal in $\mathcal{K}$, we have to calculate $\overline{E}(f - g)$ for all $g \in \mathcal{K} \setminus \{f\}$. If there exists g such that $\overline{E}(f - g) < 0$, which is equivalent to $\underline{E}(g - f) > 0$, then f is dominated by g and therefore f is not maximal. Denote $h \coloneqq g - f$. We can calculate $\underline{E}(h)$ through solving either (A1) or (B1):

$$\begin{aligned}
\text{(A1)} \qquad & \min && \sum_{\omega \in \Omega} h(\omega) p(\omega) && \text{(A1a)} \\
& \text{subject to} && \forall g_i \in \operatorname{dom} \underline{P} \colon \sum_{\omega \in \Omega} (g_i(\omega) - \underline{P}(g_i)) p(\omega) \geq 0 && \text{(A1b)} \\
& && \sum_{\omega \in \Omega} p(\omega) = 1 && \text{(A1c)} \\
& \text{where} && \forall \omega \colon p(\omega) \geq 0, && \text{(A1d)}
\end{aligned}$$

$$\begin{aligned}
\text{(B1)} \qquad & \max && \alpha && \text{(B1a)} \\
& \text{subject to} && \forall \omega \in \Omega \colon \sum_{i=1}^{n} (g_i(\omega) - \underline{P}(g_i)) \lambda_i + \alpha \leq h(\omega) && \text{(B1b)} \\
& \text{where} && \forall i \colon \lambda_i \geq 0 \quad (\alpha \text{ free}), && \text{(B1c)}
\end{aligned}$$

where $\underline{E}(h)$ is precisely the optimal value of (A1) (or (B1)). The problem (B1) is an unconditional case of the linear program in [2, p. 331]. Note that (A1) has $n + 1$ constraints and m variables. So, to determine all maximal gambles of $\mathcal{K}$, we must solve at most $k(k - 1)$ of these linear programs.

Troffaes and Hable [40, algorithm 16.4, p. 336] proposed the following strategy

for finding maximal gambles: once a non-maximal gamble is detected, it is no longer compared with other gambles. Indeed, if f is non-maximal, then there will be some gamble g that dominates f. However, if g dominates f, and f dominates h, then g will also dominate h as well. Consequently, every non-maximal gamble is dominated by at least one maximal gamble in $\mathcal{K}$. Therefore, the algorithm no longer needs to consider non-maximal gambles as soon as they are deemed non-maximal (see algorithm 8).

Algorithm 8 Find the set of maximal gambles in $\mathcal{K}$ [28, Algorithm 1]

Input: a set of k gambles $\mathcal{K} = \{f_1, \ldots, f_k\}$
Output: an index set of $\text{opt}_{\succ}(\mathcal{K})$

```
1: procedure MAXIMAL2(K)
2:     I ← ∅                                   ▷ an index set of opt_≻(K)
3:     for i = 1 : k do
4:         if ISNOTDOMINATED2(f_i, I, i) then
5:             I ← I ∪ {i}                     ▷ f_i is maximal
6:         end if
7:     end for
8:     return I
9: end procedure

10: procedure ISNOTDOMINATED2(f, I, i)
11:     for j ∈ I ∪ {i + 1, ..., k} do
12:         if E̲(f_j − f) > 0 then
13:             return False                    ▷ f is dominated by f_j
14:         end if
15:     end for
16:     return True
17: end procedure
```

Note that for algorithm 8, if the first considered gamble happens to be the only maximal gamble in $\mathcal{K}$, then the algorithm only needs to solve $2(k-1)$ linear programming problems. Specifically, to verify that none of the gambles in the set dominates the first gamble, the algorithm first needs to solve $k-1$ linear programs. Next, for each remaining gamble, the algorithm compares it with the maximal gamble, so the algorithm additionally solves $k-1$ linear programs. If all gambles in $\mathcal{K}$ are maximal, then the algorithm must solve $k(k-1)$ linear programs [40, p. 336].

We can speed up algorithm 8 by early identification of some maximal gambles in $\mathcal{K}$, for example, via E-admissibility. Specifically, if a gamble f is E-admissible in $\mathcal{K}$, then f is also maximal in $\mathcal{K}$ [12, p. 196]. Fortunately, we can simply find one of the E-admissible gambles as follows. We first sort all gambles in $\mathcal{K}$ as $f_1, \ldots, f_k$ such that for some $p \in \mathcal{M}$, for all $j > i$:

$$E_p(f_j) - E_p(f_i) \geq 0. \tag{11.2}$$

Then f_k has the highest expectation, and therefore f_k is E-admissible in $\mathcal{K}$. We can then improve algorithm 8 by initially setting $\text{opt}_{\succ}(\mathcal{K}) = \{f_k\}$. Note that if we repeat this process by sorting gambles in $\mathcal{K}$ by the expectation with respect to a different $p \in \mathcal{M}$, say $g_1, \ldots, g_k$, then we are not necessarily guaranteed to find another E-admissible gamble, as g_k may be exactly as f_k. Hence, we will use this technique only once at the beginning [28].

Note that one can early identify more maximal gambles by finding all E-admissible gambles. Troffaes and Hable [40, p. 337] recommended efficient algorithms in [16] and [43] for identifying E-admissible gambles. However, these algorithms cannot find all E-admissible gambles without solving many linear programs. Therefore, we do not apply those algorithms at the beginning.

In addition to identifying one E-admissible gamble in $\mathcal{K}$, sorting gambles with respect to the expectation also saves many comparison steps in algorithm 8 for finding $\text{opt}_{\succ}(\mathcal{K})$. Specifically, to determine whether gamble f_i is maximal in $\mathcal{K}$, we need to evaluate only $\underline{E}(f_j - f_i)$ such that $j > i$, because we immediately know that [28]:

$$\forall i < j \colon \underline{E}(f_i - f_j) \leq E_p(f_i - f_j) \leq 0. \tag{11.3}$$

An algorithm for finding maximal gambles that exploits sorting gambles is presented in algorithm 9.

Algorithm 9 Find the set of maximal gambles in $\mathcal{K}$ [28, Algorithm 2]

Input: a set of k gambles $\mathcal{K} = \{f_1, \ldots, f_k\}$ such that for some $p \in \mathcal{M}$, we have that $E_p(f_1) \leq E_p(f_2) \leq \cdots \leq E_p(f_k)$.
Output: an index set of $\text{opt}_{\succ}(\mathcal{K})$

```
1: procedure MAXIMAL3(K)
2:     I ← {k}                                   ▷ an index set of opt_≻(K)
3:     for i = 1 : k − 1 do
4:         if ISNOTDOMINATED3(f_i, i) then
5:                 I ← I ∪ {i}                   ▷ f_i is maximal
6:         end if
7:     end for
8:     return I
9: end procedure

10: procedure ISNOTDOMINATED3(f, i)
11:     for j ∈ {k, k − 1, ..., i + 1} do
12:         if E(f_j − f) > 0 then
13:             return False                      ▷ f is dominated by f_j
14:         end if
15:     end for
16:     return True
17: end procedure
```

Even though we have to do extra work to sort gambles at the beginning, we do not have to make as many comparisons in algorithm 9 as in algorithm 8. In particular, in the case that the set $\mathcal{K}$ has one maximal gamble, algorithm 9 only needs to solve $k-1$ linear programs. On the other hand, if all gambles are maximal, algorithm 9 needs to evaluate $\frac{k(k-1)}{2}$ linear programs. In both cases, they are only half of the number of comparisons of algorithm 8.

Instead of solving multiple linear programs, Jansen et al. [13] suggest to solve just a single linear program (A0) to determine whether a single gamble in $\mathcal{K}$ is

maximal or not:

$$\text{(A0)} \quad \max \quad \sum_{j=1}^{k} \sum_{\omega \in \Omega} p_j(\omega) \tag{A0a}$$

$$\text{subject to} \quad \forall j = 1, \ldots, k\colon \sum_{\omega \in \Omega} p_j(\omega) \leq 1 \tag{A0b}$$

$$\forall j = 1, \ldots, k,\ \forall g_i \in \operatorname{dom} \underline{P}\colon \sum_{\omega \in \Omega} (g_i(\omega) - \underline{P}(g_i)) p_j(\omega) \geq 0 \tag{A0c}$$

$$\forall j = 1, \ldots, k\colon \sum_{\omega \in \Omega} (f(\omega) - f_j(\omega))\, p_j(\omega) \geq 0 \tag{A0d}$$

$$\text{where} \quad \forall j = 1, \ldots, k,\ \forall \omega\colon p_j(\omega) \geq 0. \tag{A0e}$$

If the optimal value of (A0) is equal to k, then f is a maximal gamble in $\mathcal{K}$ [13]. Therefore, to determine those k gambles, we need to solve only k linear programs (see algorithm 10). However, the size of linear program is much bigger as it has $k(3+n)$ constraints and mk variables.

Note that if we modify the constraint eq. (A0b) to the following equality:

$$\forall j = 1, \ldots, k\colon \sum_{\omega \in \Omega} p_j(\omega) = 1, \tag{A0b'}$$

then (A0) is equivalent to:

$$\text{(A0')} \quad \max 0 \text{ subject to eqs. (A0b'), (A0c), (A0d) and (A0e).} \tag{11.4}$$

Therefore, if (A0') has a feasible solution, then f is a maximal gamble in $\mathcal{K}$ [13]. Note that in our simulation study, we will solve (A0') because it is in a more suitable format for the primal-dual method, as it needs fewer artificial variables [28].

We will benchmark these algorithms 8, 9 and 10 later in chapter 12.

Algorithm 10 Find the set of maximal gambles in $\mathcal{K}$ [28, Algorithm 3]

Input: a set of k gambles $\mathcal{K} = \{f_1, \ldots, f_k\}$
Output: an index set of $\mathrm{opt}_{\succ}(\mathcal{K})$

```
1: procedure MAXIMAL1(K)
2:     I ← ∅                                   ▷ an index set of opt_≻(K)
3:     for i = 1 : k do
4:         Solve linear program (A0') for f_i
5:         if (P0') has a feasible solution then
6:             I ← I ∪ {i}                     ▷ f_i is maximal
7:         end if
8:     end for
9:     return I
10: end procedure
```

11.2 Algorithms for finding interval dominant gambles

Recall that every maximal gamble in $\mathcal{K}$ is also interval dominant. Therefore, before running each algorithm in section 11.1, we can eliminate some non-maximal gambles in $\mathcal{K}$ by finding $\mathrm{opt}_{\sqsupset}(\mathcal{K})$. To check whether a gamble f is interval dominant in $\mathcal{K}$, we first calculate $\max_{g \in \mathcal{K}} \underline{E}(g)$. Then f is interval dominant if

$$\overline{E}(f) \geq \max_{g \in \mathcal{K}} \underline{E}(g). \tag{11.5}$$

Overall, to handle k gambles, we have to solve $2k-1$ linear programs [40, p. 337]. An algorithm for finding interval dominant gambles in $\mathcal{K}$ is summarized in algorithm 11.

In chapter 12, in addition to benchmarking these three algorithms for finding $\mathrm{opt}_{\succ}(\mathcal{K})$, we will also run these three algorithms applied to interval dominance. Specifically, we will run algorithm 11 at the beginning to delete non-maximal gambles in $\mathcal{K}$, and then run those three algorithms on $\mathrm{opt}_{\sqsupset}(\mathcal{K})$. This may speed up those three algorithms to find maximal gambles.

Algorithm 11 Find the set of interval dominant gambles in $\mathcal{K}$ [28, Algorithm 4]

Input: a set of k gambles $\mathcal{K} = \{f_1, \ldots, f_k\}$
Output: an index set of $\mathrm{opt}_{\sqsupset}(\mathcal{K})$

```
1: procedure IntervalDom(K)
2:     I ← ∅                                   ▷ an index set of opt_⊐(K)
3:     ℓ ← arg max_{j=1}^{k} E̲(f_j)
4:     for i ∈ {1, ..., k} \ {ℓ} do
5:         if Ē(f_i) ≥ E̲(f_ℓ) then
6:             I ← I ∪ {i}                     ▷ f_i is interval dominant
7:         end if
8:     end for
9:     return I
10: end procedure
```

11.3 Fast calculation of natural extensions inside algorithms

In this section, we exploit the fact that we only need to find the sign of $\underline{E}(g - f)$, but not its exact value, to verify whether f is dominated by g or not. So, we can speed up the process of calculating the natural extension through (A1) or (B1) as follows.

Since we minimize the objective function in (A1), the optimal value of (A1) is less than or equal to other feasible objective values. Therefore, we can stop as soon as we find a feasible solution that achieves a negative objective value, because then we know that the optimal value of (A1) is also negative. In this case, f is dominated by g, and therefore f is not maximal in $\mathcal{K}$.

Similarly, as we maximize the objective function in (B1), the optimal value of (B1) is larger than or equal to other feasible objective values. Consequently, we can stop as soon as we find a feasible solution that obtains a positive objective value, because the optimal value of (B1) is also positive. In this case, f is not dominated by g, so we must continue to compare f with other gambles in $\mathcal{K}$.

Figure 11.1 illustrates these extra stopping criteria.

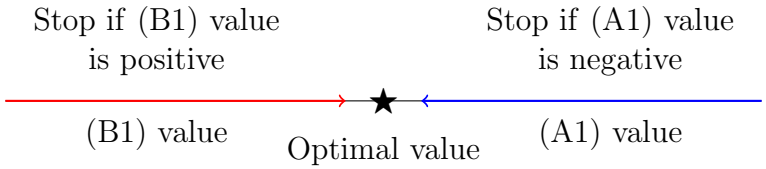

Figure 11.1: Early stopping criterion [28, Figure 1].

We can solve linear programming problems (A1) and (B1) by many methods that we mentioned before, for example, the simplex, the affine scaling or the primal-dual methods. As we know from the first contribution that the primal-dual method is particularly suitable for working with lower previsions (see chapter 9), we only consider the primal-dual method. Moreover, the primal-dual method solves both primal and dual problems simultaneously, and can therefore exploit both stopping criteria simultaneously.

Recall that, in practice, the primal-dual method can start with an arbitrary point and then generates a sequence of (not necessarily feasible) points that converges to an optimal solution [7, §7.3]. On the other hand, given initial feasible points, the method will generate a sequence of feasible points converging to an optimal solution [7, §7.3]. Therefore, to apply the early stopping criterion, we first have to find initial feasible points for (A1) and (B1). Fortunately, there is an efficient way to achieve initial feasible points for the linear programs (A1) and (B1).

For (A1), we can apply the first phase of the two-phase method to obtain an initial feasible probability mass function $p(\omega)$ as in [27, §4.2]. This technique is usually used for finding interior feasible points for the affine scaling method [7, §7.1.2]. Since the constraints of (A1) do not change, once we find a feasible starting point, we can reuse it for other problems (A1) with different objective functions. Therefore, we only need to do this once for any given lower prevision, and it is independent of the decision problem.

For (B1), we can very quickly calculate a feasible starting point without solving a linear program, using a result from Nakharutai et al. [27, Theorem 7].

Unfortunately, there is no direct way to obtain feasible starting points for the linear programming problem (A0') (otherwise we would have immediately solved the problem). Therefore, we simply solve (A0') by the standard primal-dual method.

11.4 Summary

To summarise, we studied several algorithms for finding a set of maximal gambles from the literature and proposed our algorithms. We also studied an algorithm for finding interval dominant gambles which can be applied to delete non-maximal gambles. We suggested a fast calculation to evaluate the sign of natural extensions in these algorithms.

To compare a performance of these algorithms, we are going to benchmarking them in the next chapter. Therefore, we will generate random sets of gambles such that we can pre-specify the numbers of maximal gambles and interval dominant gambles in the sets. We will perform these algorithms on those generated sets.

Chapter 12

Benchmarking and discussion

This chapter closely follows [28, §4 and §5] which we submitted it to International Journal of Approximate Reasoning on 10 December 2018. Remember that we want to compare our proposed algorithm for finding maximal gambles to two other algorithms: one proposed by Jansen et al. [13] and another proposed by Troffaes and Hable [40, p. 336]. To do so, we first provide an algorithm for generating sets of gambles with pre-specified ratios between maximal and interval dominant gamble in section 12.1. For a simulation study, we compare the performance of those algorithms on generated sets. We also present results in section 12.2 and discuss the efficiency of those algorithms for finding maximal gambles in section 12.3.

12.1 Generating sets of gambles to benchmark

As we mentioned before, we would like to generate a set of gambles $\mathcal{K}$ for benchmarking algorithms 8, 9 and 10 for finding $\operatorname{opt}_{\succ}(\mathcal{K})$ and algorithm 11 for finding $\operatorname{opt}_{\sqsupset}(\mathcal{K})$. How can we generate a set $\mathcal{K}$ such that $|\mathcal{K}| = k$, $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$ and $|\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$ where $m \leq n \leq k$?

A naive idea is to first generate $\mathcal{K} = \{g\}$, so obviously, $\operatorname{opt}_{\succ}(\mathcal{K}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}) =$

$\{g\}$. Next, we generate a gamble h such that

$$\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h\}) = \operatorname{opt}_{\succ}(\mathcal{K}) \cup \{h\} \quad \& \quad \operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}) \cup \{h\}, \tag{12.1}$$

and then we add h to $\mathcal{K}$. We repeat this process until we have $|\mathcal{K}| = |\operatorname{opt}_{\succ}(\mathcal{K})| = |\operatorname{opt}_{\sqsupset}(\mathcal{K})| = m$. After that, we generate a gamble h that satisfies

$$\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h\}) = \operatorname{opt}_{\succ}(\mathcal{K}) \quad \& \quad \operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}) \cup \{h\}. \tag{12.2}$$

Again, we add h to $\mathcal{K}$ and repeat this process until we have $|\mathcal{K}| = |\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$, whilst $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$. Finally, we generate a gamble h such that

$$\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h\}) = \operatorname{opt}_{\succ}(\mathcal{K}) \quad \& \quad \operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}), \tag{12.3}$$

and then we add h to $\mathcal{K}$. We repeat this process until we have $|\mathcal{K}| = k$ while keeping $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$ and $|\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$ as we require.

In practice, a randomly generated gamble h may not easily satisfy eq. (12.1), eq. (12.2) or eq. (12.3). We may need to sample many gambles until h satisfies the desired conditions, and therefore the process may take a while to obtain such a set $\mathcal{K}$ that we want.

Surprisingly, for any generated gamble h, we can modify h by shifting it by α, for some $\alpha \in \mathbb{R}$, so that a new gamble $h - \alpha$ meets any of the above requirements. To do so, we explain for what range of α, the gamble $h - \alpha$ satisfies either eq. (12.1), eq. (12.2) or eq. (12.3). Specifically, we identify for which values of α this results in one of the following:

(i) $\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\succ}(\mathcal{K}) \cup \{h - \alpha\}$,

(ii) $\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\succ}(\mathcal{K})$ and $\operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}) \cup \{h - \alpha\}$,

(iii) $\operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K})$.

Let $\mathcal{K}$ be a set of gambles. Given any gamble h, lemma 58 shows for which α, $h-\alpha$ is a maximal gamble in $\mathcal{K} \cup \{h-\alpha\}$.

Lemma 58. [28, Lemma 1] Let $\mathcal{K}$ be a set of gambles and let h be another gamble and $\alpha \in \mathbb{R}$. Then $h-\alpha$ is maximal in $\mathcal{K} \cup \{h-\alpha\}$ if and only if

$$\min_{f \in \text{opt}_{\succ}(\mathcal{K})} \overline{E}(h-f) \geq \alpha. \tag{12.4}$$

□

Proof. See appendix B.5. □

Lemma 58 gives an upper bound on α for which $h-\alpha$ is maximal in $\mathcal{K} \cup \{h-\alpha\}$. However, if we set α too low, then $h-\alpha$ may dominate other maximal gambles in $\text{opt}_{\succ}(\mathcal{K})$, that is, we risk having gambles f such that $f \in \text{opt}_{\succ}(\mathcal{K})$ but $f \notin \text{opt}_{\succ}(\mathcal{K} \cup \{h-\alpha\})$. The following lemma tells us how to avoid this situation.

Lemma 59. [28, Lemma 2] Let $\mathcal{K}$ be a set of gambles and let h be another gamble and $\alpha \in \mathbb{R}$. Then all maximal gambles in $\mathcal{K}$ are still maximal in $\mathcal{K} \cup \{h-\alpha\}$ if and only if

$$\max_{f \in \text{opt}_{\succ}(\mathcal{K})} \underline{E}(h-f) \leq \alpha. \tag{12.5}$$

□

Proof. See appendix B.6. □

Lemma 59 provides a lower bound on α such that $h-\alpha$ does not dominate any other maximal gambles in $\mathcal{K} \cup \{h-\alpha\}$.

Finally, by eq. (11.5), we know that $h-\alpha$ is interval dominant in $\mathcal{K} \cup \{h-\alpha\}$ if and only if

$$\overline{E}(h-\alpha) \geq \max_{f \in \mathcal{K}} \underline{E}(f), \tag{12.6}$$

which is equivalent to

$$\alpha \leq \overline{E}(h) - \max_{f \in \mathcal{K}} \underline{E}(f). \tag{12.7}$$

The next lemma ensures that these bounds on α are always ordered in the same way:

Lemma 60. [28, Lemma 3] Let $\mathcal{K}$ be a set of gambles and let h be another gamble. Then, the following holds:

$$\max_{f \in \text{opt}_{\succ}(\mathcal{K})} \underline{E}(h-f) \leq \min_{f \in \text{opt}_{\succ}(\mathcal{K})} \overline{E}(h-f) \leq \overline{E}(h) - \max_{f \in \mathcal{K}} \underline{E}(f). \tag{12.8}$$

□

Proof. See appendix B.7. □

Theorem 61 brings everything together, and summarises for which ranges of α we have that $h - \alpha$ satisfies either eq. (12.1), eq. (12.2) or eq. (12.3).

Theorem 61. [28, Theorem 1] Let $\mathcal{K}$ be a set of gambles and let h be another gamble and $\alpha \in \mathbb{R}$.

1. If we choose

$$\max_{f \in \text{opt}_{\succ}(\mathcal{K})} \underline{E}(h-f) \leq \alpha \leq \min_{f \in \text{opt}_{\succ}(\mathcal{K})} \overline{E}(h-f) \tag{12.9}$$

then

$$\text{opt}_{\succ}(\mathcal{K} \cup \{h-\alpha\}) = \text{opt}_{\succ}(\mathcal{K}) \cup \{h-\alpha\} \tag{12.10}$$

$$\text{opt}_{\sqsupset}(\mathcal{K} \cup \{h-\alpha\}) = \text{opt}_{\sqsupset}(\mathcal{K}) \cup \{h-\alpha\}. \tag{12.11}$$

2. If we choose

$$\min_{f \in \text{opt}_{\succ}(\mathcal{K})} \overline{E}(h-f) < \alpha \leq \overline{E}(h) - \max_{f \in \mathcal{K}} \underline{E}(f), \tag{12.12}$$

then

$$\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\succ}(\mathcal{K}) \tag{12.13}$$

$$\operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}) \cup \{h - \alpha\}. \tag{12.14}$$

3. If we choose

$$\alpha > \overline{E}(h) - \max_{f \in \mathcal{K}} \underline{E}(f), \tag{12.15}$$

then

$$\operatorname{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\succ}(\mathcal{K}) \tag{12.16}$$

$$\operatorname{opt}_{\sqsupset}(\mathcal{K} \cup \{h - \alpha\}) = \operatorname{opt}_{\sqsupset}(\mathcal{K}). \tag{12.17}$$

□

Figure 12.1 illustrates theorem 61. From theorem 61, algorithm 12 generates sets of k gambles $\mathcal{K}$ such that $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$ and $|\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$ where $m \leq n \leq k$.

Note that if we choose α to be much larger than $\overline{E}(h) - \max_{f \in \mathcal{K}} \underline{E}(f)$, then $h - \alpha$ can be more easily dominated.

Also note that we require the following condition for all i and j:

$$\overline{E}(h_i - h_j) < \overline{E}(h_i) - \underline{E}(h_j). \tag{12.18}$$

This ensures that there exists an α satisfying the strict inequality in eq. (12.12), because in that case, the left hand side of eq. (12.12) will be strictly less than the right hand side of eq. (12.12) (see eq. (B.50) in the appendix; the inequality there will be a strict inequality under the assumed condition). In this way, there is always an α for which $h - \alpha$ is not maximal but still interval dominant. Equation (12.18) requires that $\underline{E}$ is non-linear (i.e. genuinely imprecise), and that the gambles h_i

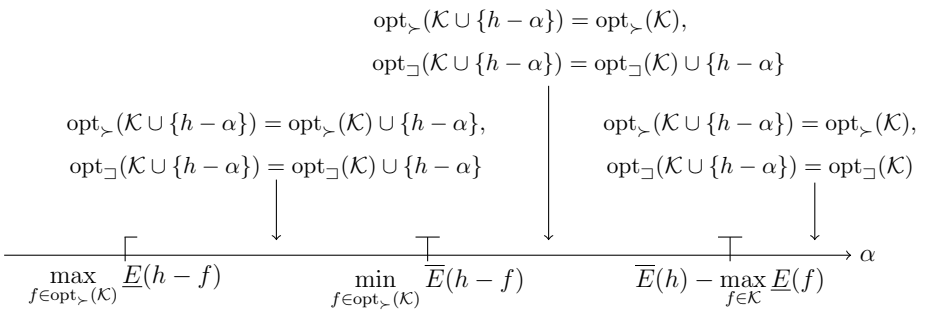

Figure 12.1: Ranges for α such that $h - \alpha$ satisfies either of the situations described in theorem 61 [28, Figure 2] .

Algorithm 12 Generate a set of k gambles $\mathcal{K}$ such that $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$ and $|\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$ where $m \leq n \leq k$ [28, Algorithm 5] .

Input: (a) Numbers m, n, k where $m \leq n \leq k$, and (b) a sequence of k gambles $h_1, \ldots, h_k$ such that $\overline{E}(h_i - h_j) < \overline{E}(h_i) - \underline{E}(h_j)$ for all $i, j \in \{1, \ldots, k\}$

Output: a set of k gambles $\mathcal{K}$ such that such that $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$ and $|\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$ where $m \leq n \leq k$

1: **procedure** GENERATESET(m, n, k)
2: $\quad\mathcal{K} \leftarrow \{h_1\}$
3: $\quad$**for** $i = 2 : m$ **do** $\quad\triangleright$ m maximal and interval dominant
4: $\quad\quad$Choose α such that $\max_{f \in \mathrm{opt}_{\succ}(\mathcal{K})} \underline{E}(h_i - f) \leq \alpha \leq \min_{f \in \mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h_i - f)$
5: $\quad\quad\mathcal{K} \leftarrow \mathcal{K} \cup \{h_i - \alpha\}$
6: $\quad$**end for**
7: $\quad$**for** $i = m + 1 : n$ **do** $\quad\triangleright$ $n - m$ interval dominant but not maximal
8: $\quad\quad$Choose α such that $\min_{f \in \mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h_i - f) < \alpha \leq \overline{E}(h_i) - \max_{f \in \mathcal{K}} \underline{E}(f)$
9: $\quad\quad\mathcal{K} \leftarrow \mathcal{K} \cup \{h_i - \alpha\}$
10: $\quad$**end for**
11: $\quad$**for** $i = m + n + 1 : k$ **do** $\quad\triangleright$ $k - (m + n)$ not interval dominant
12: $\quad\quad$Choose α such that $\alpha > \overline{E}(h_i) - \max_{f \in \mathcal{K}} \underline{E}(f)$
13: $\quad\quad\mathcal{K} \leftarrow \mathcal{K} \cup \{h_i - \alpha\}$
14: $\quad$**end for**
15: $\quad$**return** $\mathcal{K}$
16: **end procedure**

are non-constant and linearly independent. For example, if for each ω, we sample $h_i(\omega)$ uniformly from $[0, 1]$, then this requirement is practically always satisfied, and if not, we can simply re-sample h_i until it is.

Also note that algorithm 12 has to calculate many natural extensions for each new generated gamble. This is required to ensure that the numbers of maximal gambles and interval dominant gambles in the sets are exactly as specified. Otherwise, we cannot have precisely the number of maximal and interval dominant gambles as we require. Therefore, we limit the size of the set outcomes and the set of gambles, specifically, we consider $|\Omega| \leq 2^6$ and $|\mathcal{K}| \leq 2^8$.

12.2 Benchmarking results

To benchmark those algorithms 8, 9, 10 and 11 from chapter 11, in this section, we generate random sets of gambles. We consider the case that $|\Omega| = 2^2$ and $|\Omega| = 2^6$ and the number of gambles in $\mathcal{K}$ where $k = 2^j$ for $j \in \{4, 6, 8\}$. For each case, random sets of gambles $\mathcal{K}$ are generated as follows.

1. We first generate a lower prevision $\underline{P}$ that avoids sure loss. To do so, we use [27, algorithm 2] with 2^4 coherent previsions to generate a polyhedral lower prevision. Next, we use [27, algorithm 4] under this polyhedral lower prevision to obtain a lower prevision $\underline{P}$ that avoids sure loss, with $\operatorname{dom} \underline{P} = 2^4$.

2. We generate k gambles, say, $h_1, \ldots, h_k$ as follows. For each ω and i, we sample $h_i(\omega)$ uniformly from $[0, 1]$ and check whether they satisfy eq. (12.18).

3. We use algorithm 12 to generate random sets $\mathcal{K}$ such that $|\mathcal{K}| = k$, $|\operatorname{opt}_{\succ}(\mathcal{K})| = m$ and $|\operatorname{opt}_{\sqsupset}(\mathcal{K})| = n$ where $m \leq n \leq k$, where we use the previously generated $\underline{P}$ to evaluate $\underline{E}$ and $\overline{E}$. Note that in algorithm 12, we choose α in the first

loop as follows: we sample δ uniformly from $(0, 1)$, and set

$$\alpha \coloneqq \delta \max_{f \in \mathrm{opt}_{\succ}(\mathcal{K})} \underline{E}(h_i - f) + (1 - \delta) \min_{f \in \mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h_i - f). \tag{12.19}$$

For α in the second loop, we choose it as follows: sample δ uniformly from $(0, 1)$, and set

$$\alpha \coloneqq \delta \min_{f \in \mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h_i - f) + (1 - \delta) \left(\overline{E}(h_i) - \max_{f \in \mathcal{K}} \underline{E}(f) \right), \tag{12.20}$$

For the last loop, we set $\alpha \coloneqq \overline{E}(h_i) - \max_{f \in \mathcal{K}} \underline{E}(f) + \epsilon$, where we sample ϵ uniformly from $(0, 1)$.

In the simulation, we would like to cover a range of possible options of m, n, and k that satisfy $m \leq n \leq k$. Therefore, for each different size of $\mathcal{K}$, we consider 10 options that vary the number of maximal gambles m and the number of interval dominant gambles n in $\mathcal{K}$ which are illustrated in fig. 12.2 and table 12.1.

These 10 options can be grouped as follows. Options a to d represent the cases where we fix $m = 1$ while increasing n from 1 to k. Options d, g, i, and j represent the cases where we fix $n = k$ while increasing m from 1 to k. Options a, e, h, and j represent the cases where we fix $m = n$ while increasing m and n jointly from 1 to k. Finally, option f represents a case where $m < n < k$.

We then apply algorithms 8, 9 and 10 on each generated set of gambles $\mathcal{K}$. For algorithm 8 from [40, algorithm 16.4, p. 336], and our algorithm 9, we use our implementation of the primal-dual method that includes all of the improvements discussed in section 11.3 (feasible starting points and early stopping criteria). For algorithm 10, we simply solve (P0') by the primal-dual method.

To investigate whether interval dominance is helpful for finding maximal gambles, we also run each algorithm with and without algorithm 11. Specifically, we run algorithms 8, 9 and 10 as such, but additionally we also run algorithm 11 to obtain

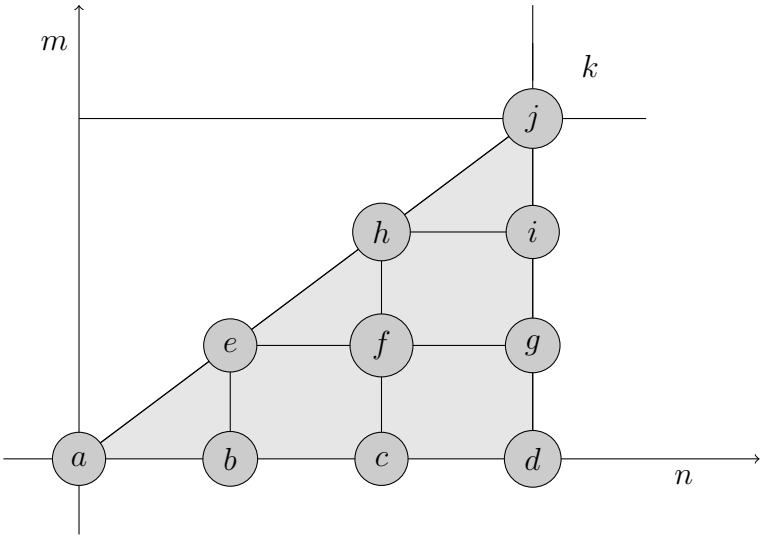

Figure 12.2: The area of $m \leq n \leq k$ and 10 options label the different m and n that we consider in the simulation (see table 12.1) [28, Figure 3].

Options	$\lvert\mathcal{K}\rvert = 2^4$		$\lvert\mathcal{K}\rvert = 2^6$		$\lvert\mathcal{K}\rvert = 2^8$	
	m	n	m	n	m	n
a	1	1	1	1	1	1
b	1	5	1	21	1	85
c	1	11	1	42	1	170
d	1	16	1	64	1	256
e	5	5	21	21	85	85
f	5	11	21	42	85	170
g	5	16	21	64	85	256
h	11	11	42	42	170	170
i	11	16	42	64	170	256
j	16	16	64	64	256	256

Table 12.1: Table of points that indicate different sizes of set $\mathcal{K}$ with vary the number of maximal gambles m and the number of interval dominant gambles n in $\mathcal{K}$ [28, Table 1].

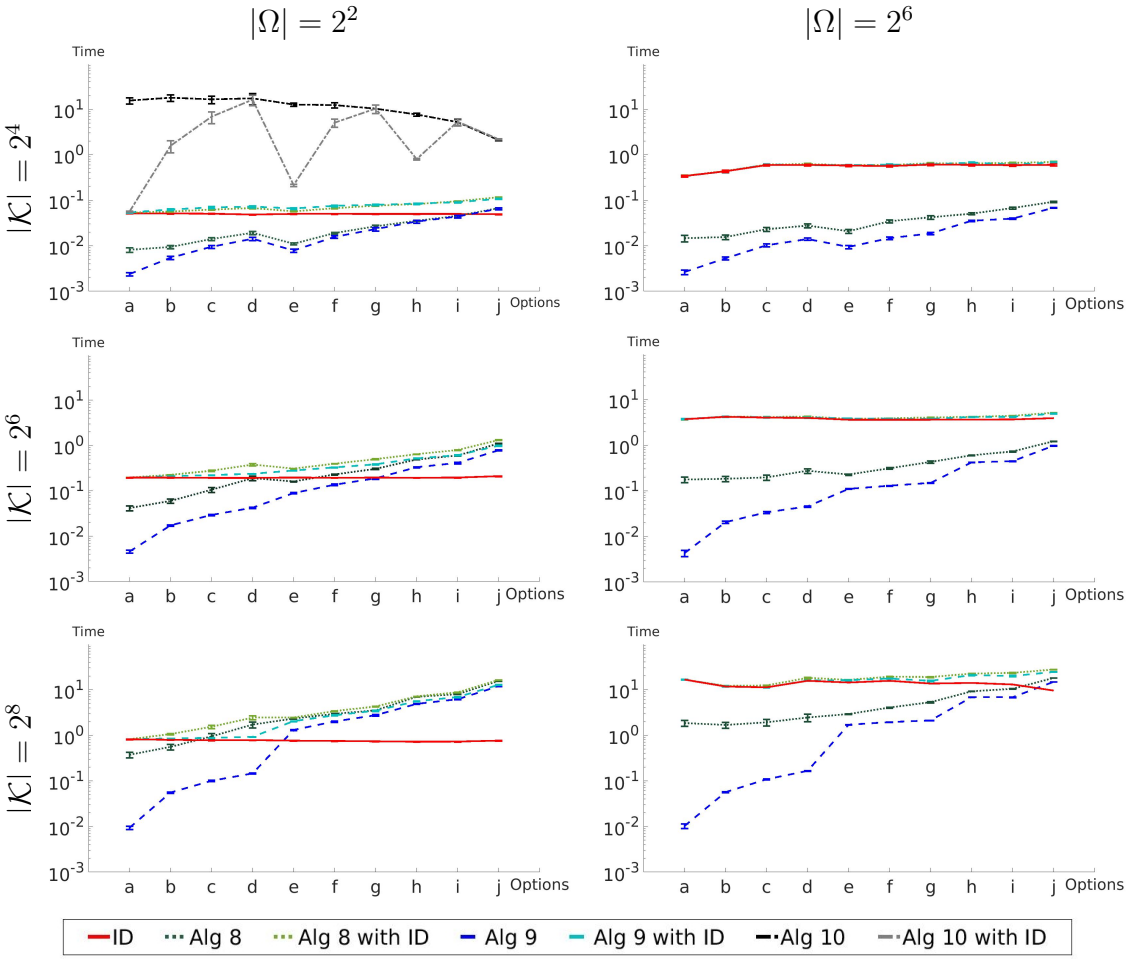

Figure 12.3: Comparison plots of the average computational time for algorithms 8, 9 and 10 for finding maximal gambles and for algorithm 11 for finding interval dominant gambles (ID). The number of outcomes in left column is 2^2 and 2^6 in the right column. Each row represents a different number of gambles with various options of the numbers of maximal gambles and interval dominant gambles in the set (see table 12.1 for each option). The labels indicate algorithms with and without algorithm 11. We fix $|\operatorname{dom} \underline{P}| = 2^4$ [28, Figure 4].

a set of interval dominant gambles, and then again run each of algorithms 8, 9 and 10 on just the resulting set of interval dominant gambles. In all cases, we measure the total computational time taken, i.e. including the time taken on algorithm 11 when applicable. Note that computational time to run algorithm 9 includes the time for sorting gambles from the lowest to the highest expectation as in eq. (11.2). To do so, we used *quicksort* which is available in MATLAB (R2016a) [22]. We repeat this process 100 times. Figure 12.3 summarizes the results.

Figure 12.3 shows the average computational time taken during each algorithm

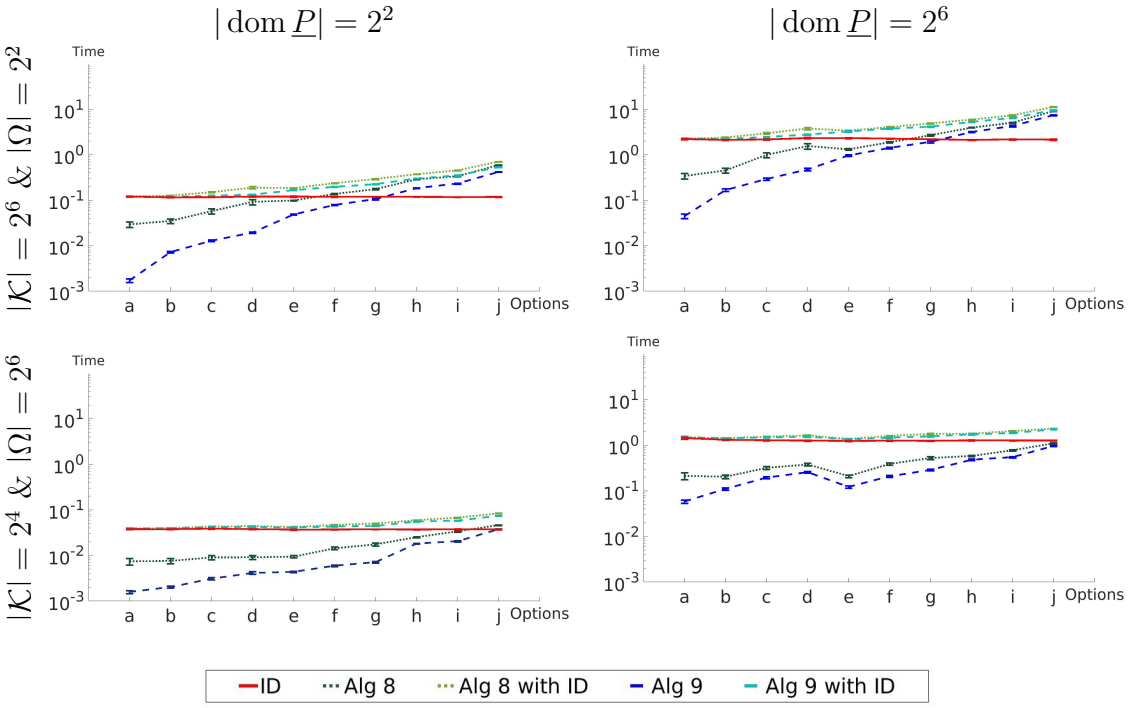

Figure 12.4: Comparison plots of the average computational time for algorithms 8 and 9 for finding maximal gambles and algorithm 11 for finding interval dominant gambles (ID). In the left column, $|\operatorname{dom}\underline{P}| = 2^2$ and in the right column, $|\operatorname{dom}\underline{P}| = 2^6$. Each row represents different numbers of gambles and outcomes with various options of the numbers of maximal gambles and interval dominant gambles in the set (see table 12.1 for each option). The labels indicate algorithms with and without algorithm 11 [28, Figure 5].

with and without algorithm 11. The average computational time taken for only algorithm 11 is also presented there. In the top left plot, we show the average computational time for algorithms 8, 9 and 10. In the remaining plots, the average computational time for algorithm 10 is so high that its performance is completely dominated by the performance of the other algorithms, and is therefore not presented in these plots. In the left column, the number of outcomes is 2^2 and in the right column, it is 2^6. Each row indicates a different size of $\mathcal{K}$.

We also consider an impact of the size of $\operatorname{dom}\underline{P}$. Figure 12.4 shows the average computational time taken for algorithms 8 and 9 with and without algorithm 11. In the left column, the number of gambles in the domain of $\underline{P}$ is 2^2 and in the right column, it is 2^6. Each row represents different numbers of outcomes and gambles.

In both figs. 12.3 and 12.4, the horizontal axis indicates different options of m, n,

and k that we consider in table 12.1. The vertical axis presents the computational time which is averaged over 100 randomly generated sets of gambles. The error bars on the figure represent approximate 95% confidence intervals on the mean computational time.

We also solved (A0) by the simplex method in algorithm 10, using the default simplex method available in MATLAB (R2016a) [22]. However, this was still slower than algorithms 8 and 9. As there is no change in general conclusion, we do not show those plots here.

12.3 Discussion

According to our numerical results, the relative performance of algorithms 8, 9 and 10 depends on (i) the numbers of outcomes and gambles in the sets, (ii) the ratios of maximal and interval dominant gambles, and (iii) the number of gambles in the domain of lower previsions. In particular, if one of these numbers is increasing, then, generally, the average computational time taken by the algorithm is longer. In contrast, the average computational time taken by algorithm 11 does not depend on the ratios of maximal and interval dominant gambles, but it depends on the numbers of outcomes and gambles in the set and the number of gambles in the domain of lower previsions. This is because algorithm 11 has to evaluate the same number of natural extensions regardless of the structure of the problem.

We observed that performing interval dominance (algorithm 11) at the beginning benefits algorithm 10 as it makes the linear program smaller, especially if there are many non-interval dominant gambles. Therefore, when using algorithm 10, we would strongly recommend to run algorithm 11 first.

In contrast, perhaps surprisingly, interval dominance (algorithm 11) does not help algorithms 8 and 9. Even though using algorithm 11 can eliminate some non-

maximal gambles, applying algorithm 11 first and then performing algorithm 8 or algorithm 9 is still slower than performing only algorithm 8 or algorithm 9. Therefore, we do not recommend applying algorithm 11 before algorithm 8 or algorithm 9.

Overall, both algorithms 8 and 9 outperform algorithm 10 by an order of magnitude. Algorithm 9 also outperforms algorithm 8 in all cases in the simulation, especially when there is only one maximal gamble in the set. In the case that the numbers of maximal gambles in the set are increasing, there is no big difference in the time taken on algorithms 8 and 9, but algorithm 9 still slightly outperforms algorithm 8. When we vary the number of gambles in the domain of lower previsions, the conclusion does not change, i.e., algorithm 9 still outperforms algorithm 8.

Given the range of our benchmarking study, we conclude that our newly proposed algorithm, algorithm 9, is a good choice for implementations, as it outperformed all other algorithms tested over all scenarios considered.

12.4 Summary

We summarise the third contribution here. In this work, we proposed a new algorithm (algorithm 9) for finding maximal gambles and compared its performance with Jansen et al. [13] (algorithm 10) and Troffaes and Hable [40, p. 336]'s algorithms (algorithm 8). We also studied the impact of applying interval dominance (algorithm 11) to eliminate non-maximal gambles as this can reduce the size of the problem.

To find the set of all maximal gambles, Jansen et al. [13]'s algorithm solves a single large linear program for each gamble, while Troffaes and Hable [40, p. 336]'s algorithm and our new algorithm solve a larger sequence of smaller linear programs. For the second case, we proposed early stopping criteria to evaluate the sign of calculating natural extensions in these algorithms. Based on earlier work [27], we applied

common feasible starting points for the entire sequence of linear programs. We found that the primal-dual method can exploit these improvements most effectively, and performs best overall.

To benchmark these algorithms, we presented a new algorithm for generating random sets of gambles with a pre-determined proportion of maximal and interval dominant gambles. This algorithm can be useful for others who want to test their algorithms. We compared computational performance of algorithms 8, 9 and 10 with and without algorithm 11 on these generated sets.

We observed that applying interval dominance benefits Jansen et al. [13]'s algorithm, but not the other two algorithms. However, Jansen et al. [13]'s algorithm is outperformed by the other two algorithms. We found that our algorithm, without using interval dominance, outperforms all other algorithms in all scenarios in our benchmarking. Therefore, we recommend our newly proposed algorithm, algorithm 9 for an implementation for finding maximal gambles.

Chapter 13

Conclusion

Several authors studied and provided algorithms for solving linear programs for checking avoiding sure loss and for finding maximal gambles. However, as far as we know, there was no comparative study and analysis of how to improve algorithms for solving these linear programs. Moreover, there was no framework for testing these algorithms. Therefore, we investigated three commonly used linear programming methods for checking avoiding sure loss and for finding maximal gambles. To set up the framework for checking avoiding sure loss, we provided algorithms for generating sets of desirable gambles that avoid or do not avoid sure loss. We also gave the algorithm for generating sets of gambles with a predetermined ratio of maximal and interval dominant gambles. This algorithm can be used to set up a framework to test algorithms for finding maximal gambles. In addition, we presented a case of checking avoiding sure loss for sets of desirable gambles derived from betting odds and a free coupon though the Choquet integral. The thesis consists of three contributions which are outlined below.

The goal of the first contribution was to investigate algorithms for efficiently solving linear programs for checking avoiding sure loss. To achieve this goal, we studied basic concepts including: linear programming problems in chapter 2 and

three algorithms, namely, the simplex, the affine scaling and the primal-dual methods in chapters 3, 4 and 5 respectively, and desirability and lower previsions in chapter 6. Based on the structure of the linear programs for checking avoiding sure loss, we slightly reduced their sizes and suggested two improvements for these three methods, that is, direct ways to calculate the feasible starting points and an early stopping criterion in chapter 7. To benchmark these improvements, in chapter 8, we gave algorithms for generating random sets of desirable gambles that either avoid or do not avoid sure loss, and conducted a comparative study for these three methods on generated sets. In chapter 9, we found that as the affine scaling and primal-dual methods benefited from the improvements, they outperformed the simplex method in most scenarios in the simulation. Consequently, the simplex method was not a good option for checking avoiding sure loss. The results suggested that in the case of small problems, the performances of all methods were similar, but if problems were large, the improved primal-dual method was at least three times faster than other methods.

Note that our algorithms for generating sets of desirable gambles that either avoid or do not avoid sure loss can be a basic framework which benefits a wide range of situations for testing algorithms for checking avoiding sure loss. Moreover, as we tested the hardest case where only a single gamble violates the condition of avoiding sure loss, any positive computational gain in these cases implied an at least as large gain for more general applied cases where more than one gamble violates consistency.

Overall, this study was the first step to improve algorithms to efficiently solve linear programs for checking avoiding sure loss. This research is not only useful to efficiently solve linear programs for checking avoiding sure loss, but also applicable to other linear programs that have a similar structure as it can reduce the effort required in the pre-solve phase of some of these linear programming algorithms, and speed up the performance of evaluating the sign of lower previsions. A further

implication was presented in the third contribution.

For the second contribution in chapter 10, we studied whether and how customers can exploit betting odds and free coupons in order to find strategies that can make a profit. To find out, we viewed betting odds and free coupons as a set of desirable gambles and checked whether such a set avoids sure loss or not. For this specific problem, unlike checking avoiding sure loss in the first contribution, we showed that we can check avoiding sure loss by evaluating the natural extension through the Choquet integral which is equal to the natural extension. If the set does not avoid sure loss, then we showed that a strategy for the customer to bet in order to make a guaranteed gain can be read from an optimal solution of the linear programs. We showed that this optimal solution can be obtained by using the Choquet integral and complementary slackness. As an illustration, we presented some actual betting odds in the market and showed that all sets of desirable gambles derived from those odds avoid sure loss. However, with a free coupon, there are some combinations of bets that the customer could make in order to get a profit.

As the natural extension in the second contribution is 2-monotone, it can be computed through the Choquet integral [39, p. 125]. The technique of applying the Choquet integral and complementary slackness to obtain an optimal solution could be applied to evaluate any natural extension which is 2-monotone. Moreover, this approach is useful to other linear programs that involve the same structure.

For the last contribution, we proposed our new algorithm for finding maximal gambles and compared it with two existing algorithms; one is proposed by Jansen et al. [13] and another by Troffaes and Hable [40, p. 336]. Based on the first contribution, we presented efficient ways to find common feasible starting points in these algorithms. We then exploited these feasible starting points to develop early stopping criteria for the primal-dual method, further improving efficiency. We found that the primal-dual method works best. Since applying interval dominance at the

beginning could reduce the size of the problems, in addition to comparing the above mentioned algorithms for finding maximal gambles, we performed these algorithms with and without applying interval dominance.

To set up a framework for benchmarking these algorithms, in chapter 12 we presented the algorithm for generating sets of gambles with pre-specified numbers of maximal and interval dominant gambles, and conducted a comparison test for these algorithms on generated sets. We found that using interval dominance at the beginning to eliminate non-maximal gambles benefits Jansen et al. [13]'s algorithm as this can make the problem smaller, but not the other two algorithms. Based on theoretical considerations, our newly proposed algorithm, algorithm 9, is a good choice for implementations as it reduced the number of linear programs, as well as the number of iterations. Our benchmarking study quantified these improvements, and we found that without using interval dominance, it outperformed all other algorithms in all cases in our simulation.

Note that in order to generate sets of gambles for benchmarking algorithms for finding maximal gambles, we are unable to increase the size of the set of outcomes and the set of gambles due to the massive computational time required. This is because the algorithm has to evaluate many natural extensions for each new generated gamble in order to ensure that the numbers of maximal gambles and interval dominant gambles in the sets are exactly as specified. This leads to a huge computational effort when the size of the outcomes and the number of the gambles in the sets are large. Because of this time consuming issue, we limited our range of the size of the outcomes and the number of the gambles to be $|\Omega| \leq 2^6$ and $|\mathcal{K}| = 2^8$. Our method for generating sets of gamble clearly has some limitations. Nevertheless we believe our method could be a framework for situations that require pre-specified numbers of maximal and interval dominant gambles. In the future work, it would be interesting if we can increase the size of the problems in our benchmarking, for example, by parallelizing the task of calculating natural extensions.

Throughout these contributions, we have managed to fill a gap in the literature and we have formed a novel framework for comparative study. We have clearly improved methods to efficiently solve linear programs for checking avoiding sure loss and to efficiently find maximal gambles. Specifically, our investigations into this area have shown a significant improvement to linear programming algorithms. For an implementation, we have demonstrated that the primal-dual method, which mostly benefited form this improvement, was a good choice to solve linear programs for lower previsions.

Results so far have been very encouraging to investigate other linear programs that have similar structures, for example, to check coherence for lower previsions or to improve algorithms for finding interval dominant gambles. Especially, to improve an algorithm for finding interval dominant gambles, we can investigate an early stopping criterion for evaluating the difference between natural extensions and direct ways to obtain feasible starting points. This approach may lead to an improved version of algorithms for finding interval dominant gambles.

Bibliography

[1] Kurt M. Anstreicher. Linear programming: Karmarkar projective algorithm. In Christodoulos A. Floudas and Panos M. Pardalos, editors, *Encyclopedia of Optimization, Second Edition*, pages 1889–1891. Springer, New York, USA, 2009.

[2] Thomas Augustin, Frank P. A. Coolen, Gert De Cooman, and Matthias C. M. Troffaes, editors. *Introduction to Imprecise Probabilities*. Wiley Series in Probability and Statistics. Wiley, 2014. ISBN 978-0-470-97381-3. URL `http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470973811.html`.

[3] George Boole. *An investigation of the laws of thought on which are founded the mathematical theories of logic and probabilities*. Walton and Maberly, London, 1854.

[4] Dominic Cortis. Expected values and variances in bookmaker payouts: A theoretical approach towards setting limits on odds. *The Journal of Prediction Markets*, 9(1):1–14, 2015.

[5] Bruno de Finetti. *Theory of Probability: A Critical Introductory Treatment*. Wiley, New York, 1974–5. Two volumes.

[6] Colantonio Emiliano. Betting markets: opportunities for many? *Annals of the University of Oradea, Economic Science Series*, 22(2):200–208, December 2013.

[7] Shu-Cherng Fang and Sarat Puthenpura. *Linear Optimization and Extensions: Theory and Algorithms*. Springer Science+Business Media New York, 1993.

[8] Shui Feng. *The Poisson-Dirichlet Distribution and Related Topics*. Springer, Berlin, Heidelberg, 2010.

[9] C.J. Goh and X.Q.Yang. *Duality in optimization and variational inequalities*. Taylor and Francis, London, 2002.

[10] Igor Griva, Stephen G. Nash, and Ariela Sofer. *Linear and Nonlinear Optimization Second edition*. SIAM, Philadelphia, 2009.

[11] Frederick S. Hillier and Gerald J. Lieberman. *Inroduction to operations research*. McGraw-Hill, 2001.

[12] Nathan Huntley, Robert Hable, and Matthias C. M. Troffaes. *Introduction to Imprecise Probabilities*, chapter Decision making, pages 190–206. Wiley, 2014. doi: 10.1002/9781118763117.ch8.

[13] Christoph Jansen, Thomas Augustin, and Georg Schollmeyer. Decision theory meets linear optimization beyond computation. In Alessandro Antonucci, Laurence Cholvy, and Odile Papini, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 329–339, Cham, 2017. Springer International Publishing. ISBN 978-3-319-61581-3.

[14] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Proceeding STOC '84 Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311, 1984.

[15] L. G. Khachiyan. Polynomial algorithm in linear algorithms in linear programming. *Soviet Math. Dokl.*, 20(1):191–194, 1979.

[16] Daniel Kikuti, Fabio G. Cozman, and Cassio P. De Campos. Partially ordered preferences in decision trees: computing strategies with imprecision in prob-

abilities. In *In IJCAI Workshop on Advances in Preference Handling*, pages 118–123, 2005.

[17] Daniel Kikuti, Fabio Gagliardi Cozman, and Ricardo Shirota Filho. Sequential decision making with partially ordered preferences. *Artificial Intelligence*, 175 (7):1346 – 1365, 2011. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2010.11.017. URL `http://www.sciencedirect.com/science/article/pii/S0004370210002067`. Representing, Processing, and Learning Preferences: Theoretical and Practical Challenges.

[18] V. Klee and G.J. Minty. How good is the simplex algorithm. *In equalityies III*, pages 159–175, 1972.

[19] M. Kojima, N. Megiddo, and S. Mizuno. A primal-dual infeasible-interior-point algorithm for linear programming. *Mathematical Programming*, 61(1):263–280, 1993. doi: 10.1007/BF01582151.

[20] David G. Luenberger and Yinyu Ye. *Linear and Nonlinear Programming, Third edition.* Springer Science+Business Media, 2008.

[21] Walter F. Mascarenhas. The affine scaling algorithm fails for step-size 0.99. *Society for Industrial and Applied Mathematics*, 7(1):34–46, February 1997.

[22] MATLAB. *version 9.0.0 (R2016a).* The MathWorks Inc., Natick, Massachusetts, 2016.

[23] I. Milliner, P. White, and D. Webber. A statistical development of fixed odds betting rules in soccer. *Journal of Gambling, Business and Economics*, 3(1): 89–99, 2009.

[24] Enrique Miranda and Gert de Cooman. *Introduction to Imprecise Probabilities*, chapter Lower prevision, pages 28–55. Wiley, 2014. doi: 10.1002/9781118763117.ch2.

[25] Renato D. C. Monteiro and Ilan Adler. Interior path following primal-dual algorithms. part i: Linear programming. *Mathematical Programming*, 44(1): 27–41, 1989. doi: 10.1007/BF01587075.

[26] N. Nakharutai, M. C. M. Troffaes, and C. C. S. Caiado. Efficient algorithms for checking avoiding sure loss. *Proceedings of Machine Learning Research*, 62: 241–252, 2017.

[27] N. Nakharutai, M. C. M. Troffaes, and C. C. S. Caiado. Improved linear programming methods for checking avoiding sure loss. *International Journal of Approximate Reasoning*, 101:293–310, October 2018. doi: 10.1016/j.ijar.2018.07.013.

[28] Nawapon Nakharutai, Matthias C.M. Troffaes, and Camila C.S. Caiado. Improving and benchmarking of algorithms for decision making with lower previsions. submitted to International Journal of Approximate Reasoning.

[29] Nawapon Nakharutai, Camila C.S. Caiado, and Matthias C.M. Troffaes. Evaluating betting odds and free coupons using desirability. *International Journal of Approximate Reasoning*, 106:128 – 145, 2019. ISSN 0888-613X. doi: https://doi.org/10.1016/j.ijar.2019.01.002. URL `http://www.sciencedirect.com/science/article/pii/S0888613X18304547`.

[30] Erik Quaeghebeur. *A Propositional CONEstrip Algorithm*, pages 466–475. Springer International Publishing, 2014. ISBN 978-3-319-08852-5. doi: 10.1007/978-3-319-08852-5_48.

[31] Erik Quaeghebeur, Chris Wesseling, Emma Beauxis-Aussalet, Teresa Piovesan, and Tom Sterkenburg. The CWI world cup competition: Eliciting sets of acceptable gambles. In Alessandro Antonucci, Giorgio Corani, Inés Couso, and Sébastien Destercke, editors, *Proceedings of the Tenth International Symposium*

on Imprecise Probability: Theories and Applications, volume 62 of *Proceedings of Machine Learning Research*, pages 277–288. PMLR, July 2017. URL `http://proceedings.mlr.press/v62/quaeghebeur17a`.

[32] I. Adler R.D. Monteiro and M.G. Resende. A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension. *Mathematics of Operations Research*, 15(2):191–214, 1990.

[33] Romesh Saigal. *Linear programming : a modem integrated analysis*. Springer Science+Business Media New York, 1995.

[34] H. A. Eiselt · C.-L. Sandblom. *Linear Programming and its Applications*. Springer-Verlag Berlin Heidelberg, 2007.

[35] Mark J. Schervish, Teddy Seidenfeld, and Joseph B. Kadane. *Some Measures of Incoherence: How Not to Gamble if You Must*. Technical Report No 660, Department of Statistics, Carnegie Mellon Univeristy, 1998.

[36] Cedric A. B. Smith. Consistency in statistical inference and decision. *Journal of the Royal Statistical Society*, B(23):1–37, 1961. URL `http://www.jstor.org/stable/2983842`.

[37] Gilbert Strang. Introduction to linear algebra. *MATLAB Central File Exchange*, 1.0, 2002, Update 30 Aug 2002. URL `https://uk.mathworks.com/matlabcentral/fileexchange/2166-introduction-to-linear-algebra?focused=5039479&tab=function`.

[38] Matthias C. M. Troffaes. Decision making under uncertainty using imprecise probabilities. *International Journal of Approximate Reasoning*, 45(1):17–29, May 2007. doi: 10.1016/j.ijar.2006.06.001.

[39] Matthias C. M. Troffaes and Gert de Cooman. *Lower Previsions*. Wiley Series in Probability and Statistics. Wiley, 2014. ISBN 978-0-470-72377-7. URL `http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470723777.html`.

[40] Matthias C. M. Troffaes and Robert Hable. *Introduction to Imprecise Probabilities*, chapter Computation, pages 329–337. Wiley, 2014. doi: 10.1002/9781118763117.ch16.

[41] T. Tsuchiya and M. Muramatsu. Global convergence of a long-step affme scaling algorithm for degenerate linear programming problems. *Research Memorandum 423, The Institute of Statistical Mathematics, 4-6-7 Minami-Azabu, Minato-ku, Tokyo 106, Japan*, January 1992; revised September, 1992.

[42] T.Terlaky and T. Tsuchiya. A note on mascarenhas' counterexample about global convergence of the affine scaling algorithm. *Applied Mathematics ans Optimization*, 40(3):287–314, 1999.

[43] Lev V. Utkin and Thomas Augustin. Powerful algorithms for decision making under partial prior information and general ambiguity attitudes. In *ISIPTA*, 2005.

[44] Robert J. Vanderbei. *Linear Programming: Foundations and Extensions, Third edition*. Springer, 2008.

[45] Nikolaos Vlastakis, George Dotsis, and Raphael N. Markellos. Beating the odds: Arbitrage and wining strategies in the football betting market. Universidad Complutense, Madrid, Spain, 2006. URL `https://bit.ly/2HhhjLJ`.

[46] Peter Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, London, 1991.

[47] Peter Walley, Renato Pelessoni, and Paolo Vicig. Direct algorithms for checking consistency and making inferences from conditional probability assessments. *Journal of Statistical Planning and Inference*, 126:119–151, 2004.

[48] Peter M. Williams. Notes on conditional previsions. Technical report, School of Math. and Phys. Sci., Univ. of Sussex, 1975.

[49] Peter M. Williams. Notes on conditional previsions. *International Journal of Approximate Reasoning*, 44(3):366–383, 2007. doi: 10.1016/j.ijar.2006.07.019.

[50] Wayne L. Winston. *Operations research: Applications and algorithms*. Duxbury press. Boston, 1987.

[51] Marco Zaffalon, Keith Wesnes, and Orlando Petrini. Reliable diagnoses of dementia by the naive credal classifier inferred from incomplete cognitive data. *Artificial Intelligence in Medicine*, 29(1–2):61–79, 2003.

Appendix A

A.1 Symbols and notations

Symbol	Description	Where defined
$\mathbb{R}$	Set of real numbers	section 2.1, p. 10
$\mathbb{N}$	Set of natural numbers	definition 18, p. 50
$c^\top$	The transpose of c	section 2.1, p. 10
I	The identity matrix	section 3.1, p. 21
Ω	Finite set of all possible outcomes	section 6.1, p. 47
ω	An element of Ω	section 6.1, p. 47
f, g, h	Gambles (mapping from Ω to $\mathbb{R}$)	section 6.1, p. 47
$\mathcal{L}(\Omega)$	Set of all gambles	section 6.1, p. 47
$\mathcal{D}$	Set of desirable gambles	section 6.1, p. 48
$\underline{P}, \overline{P}, P$	Lower/upper previsions, prevision	section 6.2, p. 50
$\operatorname{dom} \underline{P}$	The domain of $\underline{P}$	definition 19, p. 51
$\mathcal{D}_{\underline{P}}$	Set of desirable gambles of $\underline{P}$	section 6.2, p. 51
I_A	Indicator function of A	section 6.2, p.53
$\underline{p}$	Lower probability mass functions	section 6.2, p.53
$\overline{p}$	Upper probability mass functions	section 6.2, p.53
p	Probability mass function	section 6.6, p. 59
$\mathcal{E}_{\mathcal{D}}$	Natural extension of $\mathcal{D}$ as a set of gambles	definition 23, p. 54

Symbol	Description	Where defined
$\underline{E}_{\mathcal{D}}$	Natural extension of $\mathcal{D}$ as a lower prevision	definition 24, p. 54
$\overline{E}_{\mathcal{D}}$	The conjugate of $\underline{E}_{\mathcal{D}}$	section 6.3, p. 55
$\underline{E}_{\underline{P}}$	Natural extension of $\underline{P}$	definition 25, p. 55
$\underline{E}_{\overline{p}}$	Natural extension of $\underline{P}_{\overline{p}}$	section 6.4, p. 55
E_p	Expectation operator with respect to p	section 6.6, p. 59
$\Delta(\Omega)$	Set of all probability mass functions	section 6.6, p. 59
$\mathcal{M}(\underline{P})$	Credal set of $\underline{P}$	section 6.6, p. 59
$\operatorname{ext}\mathcal{M}(\underline{P})$	Set of extreme points of $\mathcal{M}(\underline{P})$	section 6.6, p. 59
$\succ$, $\sqsupset$	Strict partial orders on $\mathcal{L}(\Omega)$	section 6.7, p. 60
$\operatorname{opt}_{>}(\mathcal{K})$	Set of maximal gambles in $\mathcal{K}$ with respect to $>$	definition 39, p. 61
$\operatorname{opt}_{\succ}(\mathcal{K})$	Set of maximal gambles in $\mathcal{K}$	section 6.7, p. 61
$\operatorname{opt}_{\sqsupset}(\mathcal{K})$	Set of interval dominant gambles in $\mathcal{K}$	section 6.7, p. 61
$\operatorname{opt}_{\mathcal{M}}(\mathcal{K})$	Set of all E-admissible gambles in $\mathcal{K}$	section 6.7, p. 61
a/b	Betting odds	section 10.1, p. 109

Appendix B

B.1 Proof of corollary 28

Proof (from Appendix A in [29]). Since $A_0 = \Omega$, we can write f as

$$f = \sum_{i=1}^{n} \lambda_i I_{A_i} + \lambda_0 \tag{B.1}$$

where $\lambda_0 \in \mathbb{R}$, $\lambda_1, \ldots, \lambda_n > 0$ and $A_1 \supsetneq \cdots \supsetneq A_n \supsetneq \emptyset$. Then

$$\begin{aligned} -f &= -\sum_{i=1}^{n} \lambda_i (1 - I_{A_i^{\mathsf{c}}}) - \lambda_0 \\ &= -\sum_{i=1}^{n} \lambda_i - \lambda_0 + \sum_{i=1}^{n} \lambda_i I_{A_i^{\mathsf{c}}}. \end{aligned} \tag{B.2}$$

Therefore,

$$\overline{E}_{\overline{p}}(f) = -\underline{E}_{\overline{p}}(-f) \tag{B.3}$$

$$= -\left(-\sum_{i=1}^{n} \lambda_i - \lambda_0 + \sum_{i=1}^{n} \lambda_i \underline{E}_{\overline{p}}(A_i^{\mathsf{c}}) \right) \tag{B.4}$$

$$= \lambda_0 + \sum_{i=1}^{n} \lambda_i (1 - \underline{E}_{\overline{p}}(A_i^{\mathsf{c}}) \tag{B.5}$$

$$= \lambda_0 + \sum_{i=1}^{n} \lambda_i \overline{E}_{\overline{p}}(A_i), \tag{B.6}$$

where eq. (B.4) holds by constant additivity and comonotone additivity [39, p. 382, Prop. C.5(v)&(vii)]. □

B.2 Proof of theorem 30

Proof (from Appendix B in [29]). For the first part, suppose that $f \in \mathcal{L}(\Omega)$ and $\mathcal{D} = \{g_i, i \in \{1, \ldots, n\}\}$ is a set of desirable gambles that avoids sure loss. We find that

$$\begin{aligned} \overline{E}_{\mathcal{D}}(f) &= \inf \left\{ \alpha \in \mathbb{R} : \alpha - f \geq \sum_{i=1}^{n} \lambda_i g_i, \lambda_i \geq 0 \right\} \\ &= \min \left\{ \max_{\omega \in \Omega} \left(f(\omega) + \sum_{i=1}^{n} \lambda_i g_i(\omega) \right) : \lambda_i \geq 0 \right\}, \end{aligned} \tag{B.7}$$

where the inf is actually a min because $\mathcal{D}$ is finite. So, by lemma 29,

$$\overline{E}_{\mathcal{D}}(f) \geq 0 \iff \forall \lambda_i \geq 0, \max_{\omega \in \Omega} \left(\sum_{i=1}^{n} \lambda_i g_i(\omega) + f(\omega) \right) \geq 0. \tag{B.8}$$

For the second part, if $\mathcal{D} \cup \{f\}$ does not avoid sure loss, then $\overline{E}_{\mathcal{D}}(f) < 0$. So, by eq. (B.7), there exists an ω^* in Ω and some $\lambda_i \geq 0$ such that

$$\overline{E}_{\mathcal{D}}(f) = f(\omega^*) + \sum_{i=1}^{n} \lambda_i g_i(\omega^*) \geq f(\omega) + \sum_{i=1}^{n} \lambda_i g_i(\omega), \ \forall \omega \in \Omega. \tag{B.9}$$

Hence there is a sure loss of at least $|\overline{E}_{\mathcal{D}}(f)|$. □

B.3 Proof of theorem 42

Proof (from Appendix A in [27]). We only show that $\mathcal{D}$ avoids sure loss if and only if the optimal value of (P2) is zero since the proof that the dual problem, (D2), has feasible solutions follows immediately by theorem 12.

Firstly, by lemma 9, the optimal value of (P2) is either zero or unbounded. Next,

we show that if $\mathcal{D}$ avoids sure loss, then the optimal value of (P2) is zero, and vice versa. Note that eq. (P2b) can be written as

$$\max_{\omega\in\Omega}\left(\sum_{i=1}^{n}\lambda_i f_i(\omega)\right) \leq \sum_{i=1}^{n}\lambda_i f_i(\omega_0) + \alpha. \tag{B.10}$$

Suppose $\mathcal{D}$ avoids sure loss, then by definition 18, for all n, all $\lambda_1, \ldots, \lambda_n \geq 0$, and $f_1, \ldots, f_n \in \mathcal{D}$,

$$0 \leq \max_{\omega\in\Omega}\left(\sum_{i=1}^{n}\lambda_i f_i(\omega)\right). \tag{B.11}$$

So, by eq. (B.10),

$$0 \leq \sum_{i=1}^{n}\lambda_i f_i(\omega_0) + \alpha. \tag{B.12}$$

So, the optimal value is non-negative. Now, by putting $\lambda_i = 0$ for all i, and $\alpha = 0$, we obtain

$$\sum_{i=1}^{n}\lambda_i f_i(\omega_0) + \alpha = 0. \tag{B.13}$$

Therefore, the optimal value of (P2) is zero.

Conversely, suppose $\mathcal{D}$ does not avoid sure loss. There are non-negative $\lambda_1, \cdots, \lambda_n$ such that

$$\sup_{\omega\in\Omega}\left(\sum_{i=1}^{n}\lambda_i f_i(\omega)\right) < 0. \tag{B.14}$$

Set

$$s = \sup_{\omega\in\Omega}\left(\sum_{i=1}^{n}\lambda_i f_i(\omega)\right) \tag{B.15}$$

and choose

$$\alpha = s - \sum_{i=1}^{n}\lambda_i f_i(\omega_0). \tag{B.16}$$

Now we have $\alpha \geq 0$ and

$$\forall\omega \neq \omega_0 : \sum_{i=1}^{n}\lambda_i f_i(\omega) \leq \sum_{i=1}^{n}\lambda_i f_i(\omega_0) + \alpha = s < 0. \tag{B.17}$$

This means that

$$\sum_{i=1}^{n}\lambda_i f_i(\omega_0) + \alpha < 0 \tag{B.18}$$

is a feasible value of (P2). By lemma 9, the optimal value is unbounded. □

B.4 Proof of theorem 51

Proof (from Appendix C in [29]). Note that for each i and k, we have

$$\frac{a_{ik}}{b_{ik}} \leq \frac{a_i^*}{b_i^*} \qquad \Longleftrightarrow \qquad \frac{b_i^*}{a_i^* + b_i^*} \leq \frac{b_{ik}}{a_{ik} + b_{ik}}. \tag{B.19}$$

So,

$$\frac{b_i^*}{a_i^* + b_i^*} = \min_k \left\{ \frac{b_{ik}}{a_{ik} + b_{ik}} \right\}. \tag{B.20}$$

($\Longrightarrow$) Suppose the set of desirable gambles $\mathcal{D}$ avoids sure loss. We will show that eq. (10.17) holds. As $\mathcal{D}$ avoids sure loss, the following system of linear inequalities:

$$\forall i\colon p(\omega_i) \geq 0 \tag{B.21}$$

$$\sum_{i=1}^{n} p(\omega_i) = 1 \tag{B.22}$$

$$\forall i, k\colon \sum_{i=1}^{n} g_{ik}(\omega_i) p(\omega_i) \geq 0. \tag{B.23}$$

has a solution [46, p. 175, ll. 10–13], say $p = (p(\omega_1), \ldots, p(\omega_n))$. By lemma 50, for each i and k,

$$\frac{b_{ik}}{a_{ik} + b_{ik}} \geq p(\omega_i). \tag{B.24}$$

Then, by eq. (B.19) for each i,

$$\frac{b_i^*}{a_i^* + b_i^*} \geq p(\omega_i). \tag{B.25}$$

Therefore,

$$\sum_{i=1}^{n} \frac{b_i^*}{a_i^* + b_i^*} \geq \sum_{i=1}^{n} p(\omega_i) = 1. \tag{B.26}$$

($\Longleftarrow$) Suppose $\sum_{i=1}^{n} \frac{b_i^*}{a_i^*+b_i^*} \geq 1$ holds. Let

$$S = \sum_{i=1}^{n} \frac{b_i^*}{a_i^* + b_i^*} \quad \text{and} \quad p(\omega_i) = \frac{b_i^*}{S(a_i^* + b_i^*)}, \tag{B.27}$$

If we show that p is a feasible solution of eqs. (B.21), (B.22) and (B.23), then $\mathcal{D}$ avoids sure loss. Note that by eq. (B.27), $p(\omega_i) \geq 0$ for all i, $\sum_{i=1}^{n} p(\omega_i) = 1$ and with eq. (B.19), $\frac{b_{ik}}{a_{ik}+b_{ik}} \geq p(\omega_i)$. So, by lemma 50, $\sum_{i=1}^{n} g_{ik}(\omega)p(\omega_i) \geq 0$ holds for all g_{ik}. Therefore, p is a feasible solution of eqs. (B.21), (B.22) and (B.23) and by [46, p. 175, ll. 10–13], $\mathcal{D}$ avoids sure loss. □

B.5 Proof of Lemma 58

Proof (from Appendix A in [28]). By eq. (11.1), we have

$$h - \alpha \in \operatorname{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) \Leftrightarrow \forall g \in \mathcal{K} \colon \overline{E}(h - g - \alpha) \geq 0, \tag{B.28}$$

$$\Leftrightarrow \forall g \in \mathcal{K} \colon \overline{E}(h - g) \geq \alpha, \tag{B.29}$$

$$\Leftrightarrow \min_{g \in \mathcal{K}} \overline{E}(h - g) \geq \alpha. \tag{B.30}$$

Note that

$$\overline{E}(h - g) \geq \overline{E}(h - f) - \overline{E}(g - f). \tag{B.31}$$

Suppose that if $g \notin \operatorname{opt}_{\succ}(\mathcal{K})$, then we have $\overline{E}(g - f) < 0$ for some $f \in \operatorname{opt}_{\succ}(\mathcal{K})$ because g is dominated by at least one maximal gamble in $\mathcal{K}$ [40, p. 336]. Therefore, for all $g \notin \operatorname{opt}_{\succ}(\mathcal{K})$:

$$\exists f \in \operatorname{opt}_{\succ}(\mathcal{K}),\ \overline{E}(h - g) \geq \overline{E}(h - f). \tag{B.32}$$

Consequently,

$$h - \alpha \in \operatorname{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) \Leftrightarrow \min_{f \in \operatorname{opt}_{\succ}(\mathcal{K})} \overline{E}(h - f) \geq \alpha. \tag{B.33}$$

□

B.6 Proof of lemma 59

Proof (from Appendix B in [28]). Let f be any maximal gamble in $\mathcal{K}$. We first show that f is a maximal gamble in $\mathcal{K} \cup \{h - \alpha\}$ if and only if $\underline{E}(h - f) \leq \alpha$. We see that

$$f \in \text{opt}_{\succ}(\mathcal{K} \cup \{h - \alpha\}) \Leftrightarrow \forall g \in \mathcal{K} \cup \{h - \alpha\} \colon \overline{E}(f - g) \geq 0, \tag{B.34}$$

and since $\forall g \in \mathcal{K}$: $\overline{E}(f - g) \geq 0$,

$$\Leftrightarrow \overline{E}(f - h + \alpha) \geq 0 \tag{B.35}$$

$$\Leftrightarrow \overline{E}(f - h) \geq -\alpha \tag{B.36}$$

$$\Leftrightarrow \underline{E}(h - f) \leq \alpha. \tag{B.37}$$

Consequently, all maximal gambles in $\mathcal{K}$ are still maximal in $\mathcal{K} \cup \{h - \alpha\}$ if and only if

$$\max_{f \in \text{opt}_{\succ}(\mathcal{K})} \underline{E}(h - f) \leq \alpha. \tag{B.38}$$

□

B.7 Proof of Lemma 60

Proof (from Appendix C in [28]). We first show that

$$\forall f, g \in \text{opt}_{\succ}(\mathcal{K}) \colon \underline{E}(h - f) \leq \overline{E}(h - g). \tag{B.39}$$

holds. Let $\mathcal{K}$ be a set of gambles and let $\mathrm{opt}_{\succ}(\mathcal{K})$ be the set of maximal gambles. Suppose that h is another gamble. Then, for any $f, g \in \mathrm{opt}_{\succ}(\mathcal{K})$, we have

$$\begin{aligned} 0 &\leq \overline{E}(f-g) && \text{(B.40)}\\ &= \overline{E}(f-h+h-g) && \text{(B.41)}\\ &\leq \overline{E}(f-h)+\overline{E}(h-g) && \text{(B.42)}\\ &= -\underline{E}(h-f)+\overline{E}(h-g). && \text{(B.43)} \end{aligned}$$

Therefore, by eq. (B.43), for any $f, g \in \mathrm{opt}_{\succ}(\mathcal{K})$:

$$\underline{E}(h-f) \leq \overline{E}(h-g). \tag{B.44}$$

Consequently,

$$\begin{aligned} \max_{f\in\mathrm{opt}_{\succ}(\mathcal{K})} \underline{E}(h-f) &= \underline{E}(h-f^*) \quad \text{for some } f^* \in \mathrm{opt}_{\succ}(\mathcal{K}) && \text{(B.45)}\\ &\leq \overline{E}(h-g) \quad \text{for all } g \in \mathrm{opt}_{\succ}(\mathcal{K}) \quad \text{(by eq. (B.39))} && \text{(B.46)}\\ &\leq \min_{g\in\mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h-g). && \text{(B.47)} \end{aligned}$$

Next, we show that

$$\min_{f\in\mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h-f) \leq \overline{E}(h) - \max_{g\in\mathcal{K}} \underline{E}(g). \tag{B.48}$$

We see that

$$\begin{aligned} \min_{f\in\mathrm{opt}_{\succ}(\mathcal{K})} \overline{E}(h-f) &= \min_{g\in\mathcal{K}} \overline{E}(h-g) \quad \text{(by eq. (B.32))} && \text{(B.49)}\\ &\leq \min_{g\in\mathcal{K}} \big(\overline{E}(h) - \underline{E}(g)\big) && \text{(B.50)}\\ &= \overline{E}(h) - \max_{g\in\mathcal{K}} \underline{E}(g). && \text{(B.51)} \end{aligned}$$

□

Appendix C

C.1 Betting odds on the winner of the European Football Championship 2016

Nations	bookmakers																									
	Bet1	Bet2	Bet3	Bet4	Bet5	Bet6	Bet7	Bet8	Bet9	Bet10	Bet11	Bet12	Bet13	Bet14	Bet15	Bet16	Bet17	Bet18	Bet19	Bet20	Bet21	Bet22	Bet23	Bet24	Bet25	Bet26	Bet27
France	3	3	3	3	3	11/4	3	16/5	3	16/5	16/5	3	3	3	16/5	3	10/3	16/5	16/5	16/5	3	16/5	3	3	3	3	3
Germany	4	4	9/2	4	9/2	4	4	9/2	10/3	9/2	9/2	9/2	4	7/2	4	9/2	9/2	19/5	9/2	15/4	9/2	19/5	9/2	4	9/2	22/5	23/5
Spain	5	5	9/2	5	9/2	5	5	9/2	5	9/2	5	9/2	5	5	5	5	9/2	5	9/2	5	9/2	5	5	24/5	24/5	5	5
England	17/2	9	9	8	9	8	9	8	9	9	8	9	8	8	9	9	9	17/2	9	9	9	17/2	8	8	17/2	43/5	9
Belgium	11	10	10	10	10	11	10	11	10	10	10	10	11	10	11	11	11	9	10	9	10	9	10	10	54/5	53/5	57/5
Italy	16	16	18	16	18	16	16	16	16	16	16	18	18	16	18	16	18	16	16	14	16	16	14	17	18	89/5	91/5
Portugal	18	18	18	18	18	18	18	14	20	17	18	18	14	18	12	18	20	15	17	18	17	15	18	268/17	88/5	92/5	91/5
Croatia	25	25	22	25	22	25	25	25	20	25	25	22	25	25	25	25	25	25	25	25	25	25	22	25	26	24	27
Austria	40	40	33	33	33	40	40	40	33	40	40	40	33	33	28	40	33	40	40	33	40	40	33	40	45	43	45
Poland	50	50	50	50	50	40	40	50	50	50	40	50	50	50	50	40	50	45	50	40	50	45	50	50	47	48	50
Switzerland	66	40	66	50	66	66	50	50	66	66	50	50	50	50	50	66	66	66	66	66	66	66	50	60	66	65	64
Russia	66	66	80	66	80	80	66	66	66	80	66	50	66	66	50	66	66	66	80	66	80	66	50	66	85	84	79
Turkey	80	80	80	80	80	80	80	66	80	80	66	66	66	66	80	80	66	80	80	80	80	80	80	80	94	92	89
Wales	80	80	80	80	80	80	66	80	100	80	80	66	66	66	100	66	66	80	80	80	80	80	60	80	89	81	89
Ukraine	100	66	80	80	80	80	80	66	80	80	80	50	80	80	50	50	80	90	80	100	80	90	80	100	94	86	89
Sweden	100	80	100	80	100	100	100	100	100	100	100	100	100	80	66	100	100	100	100	100	100	100	80	100	104	90	99
Czech Rep	125	100	125	80	125	100	100	125	80	100	100	125	66	100	100	125	100	100	100	100	100	100	100	100	132	135	99
Slovakia	150	100	150	150	150	150	150	150	100	100	150	150	150	150	100	125	125	187/2	100	150	100	100	125	150	142	143	119
Rep of Ireland	150	150	150	150	150	125	150	100	150	150	150	150	100	125	100	125	125	349/4	150	150	150	112	125	150	170	156	149
Iceland	100	150	100	100	100	100	100	150	80	100	80	150	80	100	100	150	100	110	100	100	100	110	60	100	180	179	149
Romania	200	100	150	125	150	200	200	125	150	260	150	150	150	150	80	200	150	399/4	260	200	260	287/4	125	200	275	256	238
N Ireland	350	250	400	400	400	350	350	300	300	400	300	300	250	250	300	250	400	359/4	400	300	400	120	350	400	389	377	376
Hungary	350	250	400	200	400	350	350	400	350	400	250	300	250	200	200	250	250	359/4	400	250	400	359/4	250	350	541	566	79
Albania	500	250	500	400	500	350	500	400	500	500	250	200	300	250	300	400	500	363/4	500	300	500	177/4	400	500	531	513	495

Table C.1: Table of betting odds on the winner of the European Football Championship 2016 where bookmaker names are modified. Collect data from www.oddschecker.com/football/euro-2016/winner on 13-06-2016 [29, Table D9].