Games, Markets, and Online Learning

Christian Kroer

Columbia University

July 1, 2025

Preface

Why this book?

There are several great books for game theory and market design from a computational perspective. Notably, I "grew up" with the books Nisan et al. (2007), Shoham and Leyton-Brown (2008) and Easley et al. (2010), which are classic texts in this area. The idea for the present book is that it will be complementary to these books: by heavily leveraging online learning, a topic that has developed tremendously since the publication of these books, we can provide an alternative perspective on computational game theory and market design. Moreover, many of the major application results that we will cover occurred after the publication of these books, e.g. the development of the techniques used for superhuman poker AIs occurred in 2007-2018, the development of a theory of equilibrium for internet ad auctions with budgets in 2010-2020, and the application of competitive equilibrium from equal incomes for course seat allocation around 2008-2016. More generally, there has been a proliferation of ways in which online learning can be used to solve and analyze game-theoretic and market problems.

On the (Lack of) Mechanism Design in This Book

This book has an extensive amount of material on game theory and market design. Even so, one of the most fundamental concepts in market design, *mechanism design*, is only covered somewhat superficially. Mechanism design is a mathematically beautiful subject, especially the derivation of Myerson's optimal mechanism. I was tempted to add this to the present book, but I decided to keep formal mechanism design somewhat limited since it is a big topic, and not really necessary for most of the results that we cover. Moreover, there are several great books covering this topic already. Aspiring researchers would be well served by picking up one of the below books as a complement to this one.

- *An Introduction to the Theory of Mechanism Design* by Tilman Börgers. This book is a really nice read; it has an aesthetically pleasing and concise derivation of the main results.
- *Auction Theory* by Vijay Krishna. As the name suggests, this book is focused on auctions rather than general mechanism design. It also has a nice derivation of Myerson's result.

Target Audience

This book is targeted primarily at graduate students and researchers in operations research, computer science, and adjacent engineering fields. They may also provide an interesting alternative perspective for economics researchers that wish to see a computational perspective on game theory and market design. Senior undergraduate students with a background in optimization and probability should also be able to follow large parts of the book. My senior undergraduate and master's course at Columbia University uses this book, but omits topics like Blackwell approachability, fixed-point theorems, and the more advanced parts of the regret minimization chapter.

Finally, the book is also intended for practitioners. The book heavily emphasizes models and algorithms that have been deployed in practice, and thus I am hoping that it will be a useful resource. Of particular note, the book has an extensive treatment of internet advertising auctions, including the topic of budget management (usually called *pacing* or *autobidding* in the industry), which is not covered in other textbooks on algorithmic game theory.

The background requirements are as follows.

- Knowledge of linear algebra, probability, and calculus.
- A basic background in optimization:
  - (i) Linear optimization: LP modeling and LP duality
  - (ii) Convex optimization: convex sets and functions, convex duality, and KKT conditions
  - (iii) Integer optimization: basic concepts, including the ability to model a problem as an integer program.
- We will sometimes refer to basic concepts from computational complexity theory, but a background in this is not required.

The notes do not assume any background in game theory or mechanism design. For the optimization background it is not needed for every chapter, and it should be possible to learn some of these topics as you go along in the book. The first few chapters of Boyd and Vandenberghe (2004) are a good reference for convex optimization. For linear optimization, I recommend Bertsimas and Tsitsiklis (1997).

Organization of the Book

Chapter 1 gives some motivating examples of applications that can be approached with techniques from this book. Chapters 2 and 3 give a fast introduction to the most basic concepts from game theory, auctions, and mechanism design. A reader that is already familiar with these topics can skip or skim these chapters. Chapter 4 gives an introduction to online learning and regret minimization, which is a key tool in the book. That chapter also gives a constructive proof of von Neumann's minimax theorem via online learning. Chapter 5 introduces Blackwell approachability, and derives the *regret matching* algorithm. This algorithm is crucial for large-scale EFG solving. Chapter 5 can be skipped if the reader is not interested in large-scale game solving. Chapters 6 and 7 show how to solve two-player zero-sum games using regret minimization. Chapter 8 introduces extensive-form games, and show algorithmic approaches for solving large-scale EFGs. Chapter 9 introduces the *Stackelberg* equilibrium for modelling leader-follower games. Then, we introduce a special class of such games called *security games*, which model asset protection problems such as infrastructure protection and anti-poaching. Chapter 10 introduces some basic fixed-point theorems and shows how to use them to prove existence of game-theoretic equilibria and market equilibria. Chapter 11 introduces the problem of fair and efficient allocation of goods in the divisible setting. It then introduces the Fisher market equilibrium, and discusses its relationship to fair allocation. Chapter 12 introduces methods for solving large Fisher market equilibrium problems. Chapter 13 studies fair allocation when goods are indivisible. It moves on to study combinatorial utilities in such settings, and the problem of fairly allocating course seats to students. Chapter 14 introduces energy markets, the operational optimization problem that must be solved to operate the power grid, and a number of pricing approaches for achieving market equilibrium outcomes in an energy market. Chapters 15 to 18 introduce various real-world complications arising from the application of auction theory to the problem of internet advertising, including position auctions, how to handle budget constrained advertisers, and demographic fairness.

The book is meant to be readable in a largely modular fashion. For example, if a reader (with a graduate-level background in optimization or theoretical computer science) wants an introduction to fair division and competitive equilibrium in Fisher markets, they should be able to read Chapters 11 and 12 without needing to read the rest of the book. Part I, the introductory material, is used in the rest of the book to varying degrees. If the reader has no background in game theory or auctions then it is best to read this material first. A reader that is already somewhat familiar with game theory and auctions can skip these chapters and refer back to them as needed. If the reader has no background in regret minimization, then it is recommended that they read Chapter 4, as it will be used in several later chapters.

Chapters marked with a star are advanced material that can be skipped unless the reader is interested in the topic.

Acknowledgments

This book owes a large debt to several other professors that have taught courses on Economics and Computation. In particular, John Dickerson's course at UMD[1], Ariel Procaccia's course at CMU[2], and Tim Roughgarden's lecture notes (Roughgarden, 2016) and video lectures, provided inspiration for course topics as well as presentation ideas. In addition, Gabriele Farina has been instrumental in developing much of my thinking around regret minimization and learning in games through our many collaborations.

I am grateful to several former and current PhD students that have helped me teach the course at Columbia that this book is based upon. In particular, I would like to thank Rachitesh Kumar, Luofeng Liao, Aditya Shankhar Garg, and Steven Sofos DiSilvio for their help with teaching the course, which has indirectly helped shape this book as well.

In a similar vein, I would like to thank the students that have taken my course at Columbia, and provided feedback on first the lecture notes and later the book. This has tremendously improved the book.

I am grateful to the National Science Foundation and the Office for Naval Research, which have provided funding that supported my work over the years that I wrote this book.

Finally, I would also like to thank the following people for feedback on the book, and the earlier lecture notes that I based the book on: Mustafa Mert Çelikok, Jakub Černý, Darshan Chakrabarti, Ryan D'Orazio, Shuvomoy Das Gupta, Ajay Sakarwal, Felipe Verastegui-Grunewald, Eugene Vinitsky, and

[1] https://www.cs.umd.edu/class/spring2018/cmsc828m/
[2] http://www.cs.cmu.edu/ arielpro/15896s16/index.html

David Yang. A special thanks to Julien Grand-Clément, who read several parts of the book in meticulous detail and gave extensive feedback.

Any remaining errors are entirely due to the inadequacy of the author.

Contents

Notation

Basic math notation

$[n]$	The set of integers $1, \dots, n$.
$[x]^+$	Thresholding at zero, i.e. $y = [x]^+$ is such that $y_i = \max(0, x_i)$.
$\langle g, x \rangle$	The inner product $\sum_{i=1}^{d} g_i, x_i$, where d is the dimension of the vectors.
e_i	The i'th unit vector whose entries are all zero, except for the i'th entry, which is 1.
$g^\top x$	The same inner product as $\langle g, x \rangle$.
$\|x\|_1$	The ℓ_1 norm: $\|x\|_1 = \sum_{i \in [n]} \lvert x_i \rvert$
$\|x\|_2$	The ℓ_2 (or Euclidean) norm: $\|x\|_2 = \sqrt{\sum_{i \in [n]} x_i^2}$
$\|x\|_\infty$	The ℓ_∞ norm: $\|x\|_\infty = \max_{i \in [n]} \lvert x_i \rvert$
$\mathcal{P}(X)$	The *power set* of a set X, i.e. the set of all subsets of X.
$\mathbb{R}^n_{\geq 0}$	The set of nonnegative vectors in $\mathbb{R}^n$.
$\mathbb{R}^n_{\leq 0}$	The set of nonpositive vectors in $\mathbb{R}^n$.

Game notation

A_i	Action set for player i in a general-sum game.
Δ^n	The set of nonnegative vectors that sum to one, i.e. $\{x \in \mathbb{R}^n_{\geq 0} : \sum_{j=1}^{n} x_j = 1\}$.
Δ_i	The set of probability distributions over the actions A_i of player i.
X, Y	Decision sets for player 1 and player 2 in a two-player game.

Optimization notation

$D(x' \| x)$	The Bregman divergence between x' and x (see Chapter 4).
$\text{int}\, X$	The interior of a set X.
$\text{relint}\, X$	The relative interior of a set X.

PART ONE

INTRODUCTORY MATERIAL

1
Introduction and Examples

This book provides an introduction to the topics of game theory and market design, with a focus on how AI and optimization methods can be used to understand these problems, as well as enable them in practical settings. The book covers several application areas for these ideas, where each area will have real-life applications that have been deployed. A common theme underlying the areas covered by the book is that for each area, one or more of the real applications are enabled by AI and optimization. Firstly, we will repeatedly see that economic solution concepts often have some underlying convex or mixed-integer formulation of the problem, that allows us to analyze the problem via optimization theory, as well as enabling algorithms via optimization techniques. Secondly, the book uses the concept of *online learning* (also known as no-regret learning; we will use these terms interchangeably) as a unifying theme for enabling algorithms and analysis across many of the economic topics that we cover. Thirdly, applications such as poker require scaling at a level where standard optimization and online learning methods are not enough. In those settings, AI methods such as abstraction or machine learning are often used. For example, in sequential games such as poker, we may have a game that is way too large to even fit in memory. In that case, machine learning may be used to generate some coarse-grained representation of the problem. This coarse-grained representation is then typically what we solve with optimization methods. The following subsections give examples of the types of ideas and applications the book will cover.

1.1 Game Theory

The first pillar of the course will be *game theory*. In classical optimization, we have some form of objective function that we try to minimize or maximize, say $\max_{x \in X} f(x)$, where X is a convex set of possible choices, and f is some concave function. For example, perhaps we are thinking of X as a set of prices that a retailer can set for a given item, and $f(x)$ tells us the revenue that the retailer gets when setting the price x.

In game theory, on the other hand, we study settings where multiple individuals make choices, and the outcome depends on the choices of all the individuals. Suppose that we have two retailers, each choosing prices x_1 and x_2 respectively. Now, suppose that f_1 is a function that tells us the revenue received by retailer 1 in this setup. Since consumers will potentially compare the prices x_1 and x_2, we should expect f_1 to depend on both x_1 and x_2, so we let $f_1(x_1, x_2)$ be the revenue for retailer 1 generated under prices x_1 and x_2. Now we can again try to think of the optimization problem that retailer 1 wishes to solve; first let us assume that x_2 was already chosen and retailer 1 knows its value, in that case they want to solve $\max_{x_1 \in X} f_1(x_1, x_2)$. However, we could similarly argue that retailer 2 should choose their price x_2 based on the price x_1 chosen by retailer 1. Now we have a problem, because we cannot talk about optimally choosing either of the two prices in isolation, and instead we need a way to reason about how they might be chosen in a way that depends on each other. Game theory provides a formal way to reason about this type of situation. For example, the famous *Nash equilibrium*, which we will introduce below, specifies that we should find a pair x_1, x_2 such that they are mutually optimal with respect to each other. Another solution concept we will see is the *Stackelberg equilibrium*, where one retailer is assumed to go first, while anticipating the optimization problem being solved by the second retailer. From now on we will refer to each individual optimizer in a problem either as a *player* or an *agent*.

1.1.1 Nash Equilibrium

One of the most important ideas in game theory is the famous Nash equilibrium. A Nash equilibrium is a specification of an action for each player (or a probability distribution over actions) such that it is a steady state of the game, in the sense that no player wishes to change their probability distribution over actions, given the strategy of every other player. This is best illustrated with an example. Below are the payoffs of the game of rock-paper-scissors (RPS), specified as a *bimatrix* of payoffs. When specified as a bimatrix, the interpretation of the game is as follows. The set of actions for Player 1 is the rows of the matrix, and the set of actions for Player 2 is the columns of the matrix. Each entry in the bimatrix is a pair of payoffs, where the first value is the payoff to Player 1 and the second value is the payoff to Player 2. For example, if Player 1 chooses Paper (the second row) and Player 2 chooses Rock (the first column),

	Rock	Paper	Scissors
Rock	0,0	-1,1	1,-1
Paper	1,-1	0,0	-1,1
Scissors	-1,1	1,-1	0,0

Table 1.1 *The payoff matrix for Rock, Paper, Scissors.*

we get the outcome $(1, -1)$. In this outcome, Player 1 receives a payoff of 1 and Player 2 receives a payoff of -1. The goal for each player is to maximize their own payoff. A *pure* Nash equilibrium is then a pair of actions (i.e. a row and a column) such that each player is choosing a payoff-maximizing action given the choice of the other player. A *mixed-strategy* Nash equilibrium (also referred to simply as a Nash equilibrium) is a probability distribution for each player such that they maximize their own payoff given the probability distribution of the other player.

Here is an example of something that is *not* a Nash equilibrium: Player 1 always plays rock, and Player 2 always plays scissors. In this case, Player 2 is not playing optimally given the strategy of Player 1, since they could improve their payoff from -1 to 1 by switching to deterministically playing paper. In fact, this argument works for any pair of deterministic strategies, and so we see that there is no pure Nash equilibrium. Instead, RPS is an example of a game where we need randomization in order to arrive at a Nash equilibrium. Now each player gets to choose a probability distribution over their actions instead (e.g. a distribution over rows for Player 1). The value that a given player receives under a pair of mixed strategies is their expected payoff given the randomized strategies. In RPS, it's easy to see that the unique Nash equilibrium is for each player to play each action with probability $\frac{1}{3}$. Given this distribution, there is no other action that either player can switch to and improve their utility. This is what we call a (mixed-strategy) Nash equilibrium.

The famous result of John Nash from 1951 is that *every* game has a Nash equilibrium, once we allow for mixed strategies. Stated specifically for bimatrix games, the result is:

Theorem 1.1 *Every bimatrix game has a (potentially mixed-strategy) Nash equilibrium.*

In the next chapter we will see that Nash's result is broader than this. It guarantees existence for n-player games with a finite set of actions for each player, as long as we allow for mixed strategies. In Chapter 10 we will see a proof of this result, and an extension to a broader class of games.

The attentive reader may have noticed a certain redundancy in our bimatrix representation of the RPS game: the payoffs are all in the form $(1, -1)$, $(0, 0)$, and $(-1, 1)$. In all three cases, the payoff of Player 1 is exactly -1 times the payoff of Player 2. From here, we can deduce that the players have completely opposite preferences: when one player wins, the other loses. More generally, games where the sum of the two players' payoffs equal zero are called *zero-sum games*. In a two-player zero-sum game, each player can equivalently reason about minimizing the utility of the other player, rather than maximizing their own utility. Because of this special structure, zero-sum games can be represented by a single payoff matrix $A \in \mathbb{R}^{n \times m}$, where entry A_{ij} is the payoff to Player 2 when Player 1 takes the action with index i and Player 2 takes the action with index j. In this formulation, Player 1 then wishes to minimize their expected payoff over A. We will see later that this allows us to write the problem in the following form:

$$\min_{x \in \Delta^n} \max_{y \in \Delta^m} x^\top A y,$$

where $\Delta^n = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 1, x \geq 0\}$ is the probability simplex over n actions, and Δ^m is the probability simplex of m actions. Problems of this form are variously known as *matrix games*, two-player zero-sum games, or more broadly as *bilinear saddle-point problems*. The key here is that we can now represent the outcome that the players respectively want to minimize and maximize as a single objective, thereby allowing us to write the problem as a nested minimization and maximization problem (we will see later that this ordering is not important, and it can equivalently be written as a nested maximization and minimization problem). Zero-sum matrix games are very special: they can be solved in polynomial time with a linear program (LP) whose size is linear in the matrix size.

Rock-paper-scissors is of course a rather trivial example of a game. A more exciting application of zero-sum games is to use it to compute an optimal strategy for two-player poker (AKA heads-up poker). In fact, this was the foundation for many recent "superhuman AI for poker" results, as we shall discuss later. In order to model poker games we will need a more expressive game class called *extensive-form games* (EFGs). These games are played on trees, where players may sometimes have groups of nodes, called *information sets*, that they cannot distinguish among. An example is shown in Figure 1.1.

EFGs can also be represented as a bilinear saddle-point problem:

$$\min_{x \in X} \max_{y \in Y} x^\top A y,$$

where X, Y are no longer probability simplexes, but more general convex polytopes that encode the sequential decision spaces of each player. This is called

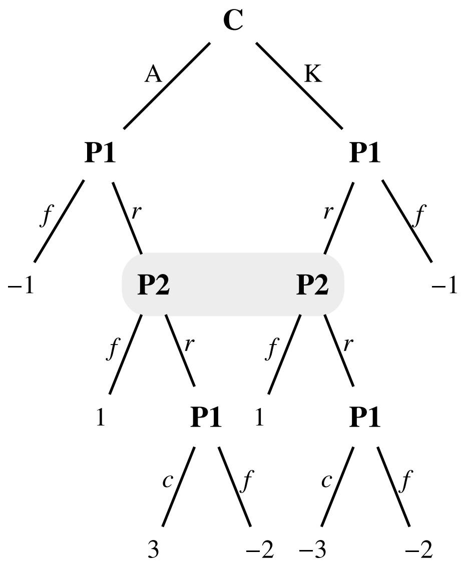

Figure 1.1 A poker game where P1 is dealt Ace or King. "r," "f," and "c" stands for raise, fold, and check respectively. Leaf values denote P1 payoffs. The shaded area denotes an information set: P2 does not know which of these nodes they are at, and must thus use the same strategy in both.

the *sequence-form* representation, covered in Chapter 8. Like matrix games, zero-sum EFGs can be solved in polynomial time with linear programming, with an LP whose size is linear in the game tree.

It turns out that in many practical scenarios, the LP for solving a zero-sum game ends up being far too large to solve. This is especially true for EFGs, where the game tree can quickly become extremely large because the LP size grows exponentially in the depth of the game. Instead, iterative methods are used in practice. What is meant by iterative methods here is the class of algorithms that build a sequence of strategies $x_0, x_1, \ldots, x_T, y_0, y_1, \ldots, y_T$ using only some form of oracle access to Ay and $A^\top x$ (this is different from writing down A explicitly!). Typically, in such iterative methods the average of the sequence of strategies $\bar{x}_T = \frac{1}{T}\sum_{t\in[T]} x_t, \bar{y}_T = \frac{1}{T}\sum_{t\in[T]} y_t$ converges to a Nash equilibrium. The reason these methods are preferred is two-fold. First, by never writing down A explicitly we save a lot of memory (now we just need enough memory to store the much smaller x, y strategy vectors). Secondly, they avoid the expensive matrix inversions involved in the typical algorithms for solving LPs such as the simplex algorithm and interior-point methods.

The algorithmic techniques we will learn for Nash equilibrium computation are largely centered around iterative methods. First, we will do a quick introduction to online learning and online convex optimization. We will learn about two classes of algorithms: 1) Methods that converge to an equilibrium at a rate of $O(1/\sqrt{T})$. These roughly correspond to saddle-point variants of gradient-descent-like methods. 2) Methods that converge to the solution at a rate of $O(1/T)$. These roughly correspond to saddle-point variants of accelerated gradient methods. Then we will also look at the practical performance of these algorithms. Here we will see that the following quote is very much true:

In theory, theory and practice are the same. In practice, they are not.

In particular, the preferred method in practice is the CFR$^+$ algorithm and later variations, all of which have a theoretical convergence rate of $O(1/\sqrt{T})$. The methods that converge at a rate of $O(1/T)$ in theory are actually slower than CFR$^+$ for most real games!

Being able to compute an approximate Nash equilibrium with iterative methods is only one part of how superhuman AIs were created for poker. In addition, abstraction and deep learning methods were used to create a small enough game that can be solved with iterative methods. We will also cover how these methods are used.

Killer applications of zero-sum games include poker (as we saw), other recreational two-player games, and generative-adversarial networks (GANs). Other applications that are, as yet, less proven to be effective in practice are robust sequential decision-making (the adversary represents uncertainty), security scenarios where we assume the world is adversarial, and defense applications.

1.1.2 Stackelberg Equilibrium

A second game-theoretic solution concept that has had extensive application in practice is what's called a *Stackelberg equilibrium*. We will primarily study Stackelberg equilibrium in the context of what is called *security games*.

Imagine the following scenario: we are designing the patrol schedule for national park rangers that try to prevent poaching of endangered wildlife in the park (such as rhinos, which are poached for their horns). There are 20 different watering holes that the rhinos frequent. We have 5 teams of guards that can patrol watering holes. How can we effectively combat poaching? If we come up with a fixed patrol schedule then the poachers can observe us for a few days and learn our schedule exactly. Afterwards they can strike at a waterhole that is guaranteed to be empty at some particular time. Thus, we need to design a schedule that is unpredictable, but which also accounts for the fact that some watering holes are more frequented by rhinos (and are thus higher value), travel constraints, etc.

In the security games literature, the most popular solution concept for this kind of setting is the Stackelberg equilibrium. In a Stackelberg equilibrium, we assume that we, as the leader (e.g. the park rangers), get to commit to our (possibly randomized) strategy first. Then, the follower observes our strategy and best responds. This turns out to yield the same solution concept as Nash equilibrium in zero-sum games, but in general games it leads to a different solution concept.

However, if we want to help the park rangers design their schedules then we will need to be able to compute Stackelberg equilibria of the resulting game model. Again, we will see that optimization is one of the fundamental pillars of the field of security games research. A unique feature of security games is that the strategy space of the leader is typically some combinatorial polytope (e.g. a restriction on the *transportation polytope*), and the problem of computing a Stackelberg equilibrium is intimately related to optimization over the underlying polytope of the defender. Because of this combinatorial nature, security games often end up being much harder to solve than zero-sum Nash equilibrium. Therefore, our focus in the security games section will be on combinatorial approaches to this problem, such as mixed-integer programming and decomposition.

Killer applications of Stackelberg games are mainly in the realm of security. They have been applied in infrastructure security (airports, coast guard, air marshals), to protect wildlife, and to combat fare evasion. A nascent literature is also emerging in cybersecurity. Outside the world of security, Stackelberg games are also used to model things like first-mover advantage in the business world.

1.2 Market Design

The second pillar of the book will be *market design*. In market design we are typically concerned with how to design the rules of the game, and how to do that in order to achieve "good" outcomes. Thus, game theory is a key tool in market design, since it will be our way of understanding what outcomes we may expect to arise, given a set of market rules.

Market design is a huge area, and so it has many killer applications. The ones we will see in this course include Internet ad auctions and how to fairly assign course seats to students. There are many others, such as how to price and assign drivers to passengers at Lyft/Uber, how to assign NYC kids to schools, how to

	Course A	Course B
Student 1	5	5
Student 2	2	8

Table 1.2 *Example course matching valuations.*

enable nationwide kidney exchanges, and how to allocate the radio spectrum efficiently.

Imagine that we are designing a mechanism for managing course enrollment. How should we decide which students get to take which courses? Certain courses are likely to have significantly more demand for seats than there are sets (e.g. machine learning courses were heavily overdemanded in the 2010s). Overall, we would like the system to somehow be *efficient*, but what does that mean? We would also like the system to be *fair*, but it's not entirely clear what that means either.

At a loss for ideas, we come up with the following solution: we will just have a sign-up system where students can sign up until a course fills up. Once a course fills up, we put other students on a waitlist that we clear on a first-in first-out basis as seats become available. The reader may find this system familiar, as it is used at many universities to manage course enrollment. Is this a good system? Well, let's look at a simple example: we will have 2 students and 2 courses, each course having 1 seat. Students are allowed to take at most one course. Let's say that each student values the courses as follows:

Student 1 arrives first, is indifferent between the courses, and arbitrarily signs up for course B. Then Student 2 arrives and signs up for A. The total *welfare* of this assignment is $5 + 2 = 7$. This does not seem to be an efficient use of resources: we can improve our solution by swapping the courses, since Student 1 gets the same utility as before, and Student 2 improves their utility. This is what's called a *Pareto-improving* allocation because each student is at least as well off as before, and at least one student is strictly better off. One desideratum for efficiency is that no such improvement should be possible; an allocation with this property is called *Pareto efficient*.

Let's look at another example. Now we have 2 students and 4 courses, where each student takes 2 courses. Again courses have only 1 seat. Now say that Student 1 shows up first, and signs up for A and B. Then Student 2 shows up and signs up for C and D. Call this assignment A_1. Here we get that A_1 is Pareto efficient, as there is no way to improve the utility of Student 2 without hurting the utility of Student 1. But it does not seem like a fair assignment. A fairer solution would be that each student gets a course with value 10 and a course

	Course A	Course B	Course C	Course D
Student 1	10	10	1	1
Student 2	10	10	1	1

Table 1.3 *Example course matching valuations.*

with value 1, for a total utility of 11 each. Let A_2 be such an assignment. One way to look at this improvement is through the notion of *envy*: each student should like their own course schedule at least as well as that of any other student. Under A_1 Student 2 envies Student 1 by a value of 18, whereas under A_2 no student envies the other. An allocation where no student envies another student is called *envy-free*. Fairness turns out to be a complicated idea, and we will see later that there are several appealing notions that we may wish to strive for.

Instead of first-come-first-serve, we can use ideas from market design to get a better mechanism. The solution that we will learn about is based on a fake-money market: we give every student some fixed budget of fake currency (aka funny money). Then, we treat the assignment problem as a market problem under the assigned budgets, and ask for what is called a *market equilibrium*. Briefly, a market equilibrium is a set of prices, one for each item, and an allocation of items to buyers. The allocation must be such that every item is fully allocated (or has a price of zero), and every buyer is getting an assignment that maximizes their utility over all the possible assignments they could afford given the prices and their budget. Given such a market equilibrium, we then take the allocation from the equilibrium, throw away the prices (the money was fake anyway!), and use that to perform our course allocation. This turns out to have a number of attractive fairness and efficiency properties. Course-selection mechanisms based on this idea are deployed at several business schools such as Columbia Business School, the Wharton School at University of Pennsylvania, Rotman School of Management at University of Toronto, and Tuck School of Business at Dartmouth.

Of course, if we want to implement this protocol we need to be able to compute a market equilibrium. This turns out to be a rich research area: in the case of what is called a *Fisher market*, where each agent i has a linear utility function $v_i \in \mathbb{R}^m_{\geq 0}$ over the m items in the market, and the items are divisible, there is a beautiful convex program that results in a market equilibrium:[1]

$$
\begin{aligned}
\max_{x \geq 0} \quad & \sum_i B_i \log(v_i^\top x_i) \\
\text{s.t.} \quad & \sum_{i \in [n]} x_{ij} \leq 1, \forall j \in [m].
\end{aligned}
$$

Here x_{ij} is how much buyer i is allocated of item j. Notice that we are simply maximizing the budget-weighted logarithmic utilities, with no prices! It turns out that the prices are the dual variables on the supply constraints. We will see some nice applications of convex duality and Fenchel conjugates in deriving this relationship. We will also see that this class of markets have a relationship to the types of auction systems that are used at large internet companies such as Google and Meta for allocating ad slots.

In the case of markets such as those for course seats, the problem is computationally harder and requires combinatorial optimization. Current methods use a mixture of MIP and local search.

[1] Note the implicit domain constraint that $v_i^\top x_i > 0$ for each i. It is usually assumed without loss of generality that each agent i has strictly positive value for at least one item.

2
Nash Equilibrium Introduction

In this section we begin our study of Nash equilibrium by giving the basic definitions, Nash's existence result, and briefly touch on computability issues. Then we will make a few observations specifically about zero-sum games, which have much more structure to exploit.

2.1 General-Sum Games

A *normal-form game* consists of:

- A set of players $N = \{1, \dots, n\}$.
- A set of actions $A = A_1 \times A_2 \times \cdots \times A_n$.
- A utility function $u_i : A \to \mathbb{R}$.

A vector $a \in A$ is called a *strategy profile*, and it denotes an action choice for every player. We will use the shorthand a_{-i} to denote the subset of a strategy vector a that does not include player i's action. As an aside, game theory often uses both the term "strategy" and "action" to refer to the course of action taken by a player. We will use "strategy" to mean a specification of what the player does at every decision point, whereas an action is what the player does at a particular decision point. Because normal-form games have only a single decision point, a strategy and an action are the same thing.

As a first solution concept we will consider *dominant-strategy equilibrium* (DSE). In DSE, we seek a strategy profile $a \in A$ such that each action a_i is a best response *no matter what a_{-i} is*. Formally, a is a DSE if for all players $i \in [n]$, alternative actions $a'_i \in A_i$ for player i, and all possible strategy profiles $a'_{-i} \in A_{-i}$ over the remaining players, it holds that $u_i(a_i, a'_{-i}) \geq u_i(a'_i, a'_{-i})$. This is a very strong property, and DSE may not exist in many games. A classic example of DSE is the *prisoner's dilemma*: two individuals are on trial for

	Silent	Confess
Silent	-1,-1	-9,0
Confess	0,-9	-6,-6

Table 2.1 *The payoff matrix for Prisoner's Dilemma.*

		Students	
		Listen	Sleep
Prof.	Prepare	$10^6, 10^6$	-10,0
	Slack off	0,-10	0,0

Table 2.2 *The payoff matrix for The Professor's Dilemma.*

a crime, and they both have two actions: "stay silent" and "confess." If both individuals stay silent then they will each get 1 year in prison. If one individual confesses and the other does not, then the confessor gets no time, but their co-conspirator gets 9 years. Finally, if both individuals confess then they both get 6 years. In this game, confessing is a DSE: it yields greater utility than staying silent no matter what the other player does. A DSE rarely exists in practice when given a game, but it can be useful in the context of mechanism design, where we get to design the rules of the game, potentially in order to induce a DSE. It is the idea underlying e.g. the second-price auction, which we will cover later.

Consider some strategy profile $a \in A$. We say that a is a *pure-strategy Nash equilibrium* if for each player i and each alternative action $a'_i \in A_i$:

$$u_i(a) \geq u_i(a'_i, a_{-i}),$$

where, again, a_{-i} denotes all the actions in a except that of i. A DSE is always a pure-strategy Nash equilibrium, but not vice versa. Consider the *Professor's dilemma*,[1] where the professor chooses a row strategy and the students choose a column strategy: In this game there is no DSE, but there's clearly two pure-strategy Nash equilibria: the professor prepares and students listen, or the professor slacks off and students sleep. But these have quite different properties. Thus, if we hope to use PNE as a prescriptive tool for what will happen, then we need to decide on which PNE will be played. This is called the *equilibrium selection problem*, and it is a major issue in general-sum games. There are at least two reasons for this: first, if we want to predict the behavior of players then how do we choose which equilibrium to predict? Second, if we want to

[1] Example borrowed from Ariel Procaccia's slides

prescribe behavior for an individual player, then we cannot necessarily suggest that they play some particular strategy from a Nash equilibrium, because if the other players do not play the same Nash equilibrium then it may be a terrible suggestion (for example, suggesting that the professor plays "Prepare" from the Prepare/Listen equilibrium, when the students are playing the Slack off/Sleep equilibrium would be bad for the professor).

Moreover, pure-strategy equilibria are not even guaranteed to exist, as we saw in the previous section with the rock-paper-scissors example.

To fix the existence issue we may consider allowing players to randomize over their choice of strategy. Let $\Delta_i = \left\{\sigma \in \mathbb{R}_{\geq 0}^{|A_i|} : \sum_{j=1}^{|A_i|} \sigma_j = 1\right\}$ be the set of possible probability distributions over the actions A_i of player i. Let $\sigma_i \in \Delta_i$ denote player i's strategy; we call σ_i a *mixed strategy*. Now we say that a strategy profile is a collection of mixed strategies, one for each player, and denote it by $\sigma = (\sigma_1, \ldots, \sigma_n)$. By a slight abuse of notation we may rewrite a player's utility function as

$$u_i(\sigma) = \sum_{a \in A} u_i(a) \prod_i \sigma_i(a_i).$$

Definition 2.1 A (mixed-strategy) Nash equilibrium is a strategy profile σ such that for all alternative strategies $\sigma_i' \in \Delta_i$:

$$u_i(\sigma) \geq u_i(\sigma_{-i}, \sigma_i').$$

An ϵ-Nash equilibrium is as above, but where the condition is relaxed to

$$u_i(\sigma) \geq u_i(\sigma_{-i}, \sigma_i') - \epsilon.$$

In Definition 2.1 we required a comparison to any $\sigma_i' \in \Delta_i$ for simplicity of notation. It is easy to show that it is enough to satisfy the inequality for every pure strategy (i.e. every strategy that puts probability one on a single action).

Nash's theorem asserts that a Nash equilibrium is guaranteed to exist (see Chapter 10 for a proof).

Theorem 2.2 *Any game with a finite set of strategies and a finite set of players has a mixed-strategy Nash equilibrium.*

Since our goal is to prescribe or predict behavior, we would also like to be able to compute a Nash equilibrium. Unfortunately this turns out to be computationally difficult even with only two players: The problem of computing a Nash equilibrium in a two-player general-sum finite game is PPAD-complete. We won't go into detail on what the complexity class PPAD is for now, but suffice it to say that it is weaker than the class of NP-complete problems, but still believed to take exponential time in the worst case. Section 2.2 gives a quick overview of the class of hardness problems that encapsulate the difficulty of computing Nash equilibrium.

As an aside, one may make the following observation about why Nash equilibrium does not "fit" in the class of NP-complete problems: typically in NP-completeness we ask questions such as "does there exist a satisfying assignment to this Boolean formula?" or "does there exist a solution to this MIP that achieves objective value at least c?" But given a particular game, we already know that a Nash equilibrium exists. Thus, we cannot ask the type of existence questions typically used in NP-complete problems; we already know the answer. Instead, it is only the task of *finding* one of the solutions that is difficult. This can be a useful notion to keep in mind when encountering other problems that have guaranteed existence. That said, once one asks for additional properties such as "does there exist a Nash equilibrium where the sum of utilities is at least v?" one gets an NP-complete problem.

Given a strategy profile σ, we will often be interested in measuring how "happy" the players are with the outcome of the game under σ. Most commonly, we are interested in the *social welfare* of a strategy profile (and especially for equilibria). The social welfare is the expected value of the sum of the player's utilities:

$$\sum_{i=1}^{n} u_i(\sigma) = \sum_{i=1}^{n} \sum_{a \in A} u_i(a) \prod_{i'=1}^{n} \sigma_{i'}(a_{i'}).$$

We already saw in the Professor's Dilemma that there can be multiple equilibria with wildly different social welfare: when the professor slacks off and the students sleep, the social welfare is zero; when the professor prepares and the students listen, the social welfare is $2 \cdot 10^6$.

2.1.1 Zero-Sum Games

In the special case of a two-player zero-sum game, we have $u_1(a) = -u_2(a) \forall a \in A$. In that case, we can represent our problem as the bilinear saddlepoint problem (BSPP) we saw in Chapter 1. The reduction to a BSPP is as follows. Suppose we have a strategy profile $\sigma = (\sigma_1, \sigma_2)$. Then we can write the expected utility of player 2 as

$$u_2(\sigma) = \sum_{a \in A} u_2(a)\sigma_1(a_1)\sigma_2(a_2).$$

This expression is a bilinear form in σ_1 and σ_2 (meaning that it is linear in σ_i for a fixed σ_{-i}). A standard fact from linear algebra is that for a fixed coordinate representation (say the standard basis), a bilinear form has an associated matrix A representing the expression. Suppose we let $x \in \Delta^n$ denote the vector corresponding to σ_1 in the standard basis, and $y \in \Delta^m$ denote the vector representing σ_2. Then the payoff to player 2 is $\langle x, Ay \rangle$, where A is the matrix with entries $A_{ij} = u_2(a_i, a_j)$ for the pair of actions $a_i \in A_1, a_j \in A_2$. Due to the zero-sum property, player 1 maximizes their utility by minimizing $\langle x, Ay \rangle$. Now suppose that player 1 plays chooses a mixed strategy $x \in \Delta^n$ under the assumption that player 2 will observe their strategy x, and *best respond* to x. Then player 1 should solve the following bilinear saddlepoint problem:

$$\min_{x \in \Delta^n} \max_{y \in \Delta^m} \langle x, Ay \rangle.$$

A first observation one may make is that the minimization problem faced by the x-player is a convex minimization problem, since the max operation is convexity preserving. This suggests that we should have a lot of algorithmic options to use. For example, we immediately see that if we run subgradient descent on the minimization problem, then we can use the optimal response of player 2 to x as a subgradient for the minimization problem. This is a very natural algorithm to use, but we will see much more numerically performant methods in Part TWO.

In fact, we have the following stronger statement, which is essentially equivalent to LP duality:

Theorem 2.3 (von Neumann's minimax theorem) *Every two-player zero-sum game has a unique value $v \in \mathbb{R}$, called the* value of the game*, such that*

$$\min_{x \in \Delta^n} \max_{y \in \Delta^m} \langle x, Ay \rangle = \max_{y \in \Delta^m} \min_{x \in \Delta^n} \langle x, Ay \rangle = v.$$

We will prove a more general version of this theorem when we discuss regret minimization in Chapter 4.

Theorem 2.3 tells us that Nash equilibria must be solutions to the min-max and max-min problems for each player. This is a very powerful property, because it allows us to compute a Nash equilibrium by solving a convex optimization problem. In fact, we can compute a Nash equilibrium in polynomial time using linear programming (LP). This reduction is obtained by dualizing the inner problem (say the maximization problem in the min-max formulation). This yields the following LP (which yields a Nash equilibrium strategy x^* for player one; to get a Nash equilibrium strategy y^* we must solve the symmetric LP where we move the min on the inside and dualize it):

$$\begin{aligned} &\min_{x,v} v \\ &\text{s.t. } v \cdot \vec{1} \geq A^\top x \\ &\quad x \in \Delta^n. \end{aligned}$$

Because zero-sum Nash equilibria are min-max solutions, they are the best that a player can do, given a worst-case opponent. Moreover, if the opponent is *not* a worst-case opponent (i.e. not best responding to our min-max strategy x^*), then a min-max solution x^* gets at least a value v, and may do even better. Conversely, any strategy x that is *not* a min-max solution is guaranteed to do worse than v against an opponent that best responds to x. These considerations are the rationale for saying that a given two-player zero-sum game has been *solved* if a Nash equilibrium has been computed for the game. Some games are trivially solvable, e.g. in rock-paper-scissors we know that the uniform distribution is the only equilibrium. However, this notion has also been applied to large games such as heads-up limit Texas hold'em, one of the smallest poker variants played by humans (which is still a huge game). In 2015, that game was *essentially solved*. The idea of *essentially solving* a game is as follows: we want to compute a strategy that is statistically indistinguishable from a Nash equilibrium in a lifetime of human-speed play. The statistical notion was necessary because the solution was computed using iterative methods that only converge to an equilibrium in the limit (but in practice get quite close very rapidly). The same argument is also used in constructing AIs for even larger two-player zero-sum poker games where we can only try to approximate an equilibrium.

Note that this min-max guarantee of Nash equilibria does not hold in general-sum games. In general-sum games, we have no payoff guarantees if our opponent does not play their part of the same Nash equilibrium that we play. Interestingly, the AI and optimization methods developed for two-player zero-sum poker turned out to still outperform top-tier human players in 6-player no-limit Texas hold'em poker, in spite of these equilibrium selection issues. An AI based on these methods ended up beating professional human players, in spite of the methods having no guarantees on performance, nor even of converging to a general-sum Nash equilibrium.

Another interesting property of zero-sum Nash equilibrium is that they are *exchangeable*: if you take an equilibrium (x, y) and another equilibrium (x', y') then (x, y') and (x', y) are also equilibria. This is easy to show from the minimax formulation.

2.2 Complexity of Computing Nash Equilibrium in General-Sum Games

As stated earlier, computing a Nash equilibrium in a general-sum game is conjectured to be a hard problem, in the sense that there is no polynomial-time algorithm for computing an equilibrium. This subsection gives a very brief overview of what hardness means in the context of Nash equilibrium computation.

The hardness of computing a Nash equilibrium is studied with the class of problems called PPAD (Polynomial Parity Argument, Directed version) from computational complexity theory. PPAD is a collection of computational problems. A problem is *PPAD-hard* if it is at least as difficult as any other problem in PPAD, meaning that every problem in PPAD can be reduced to the PPAD-hard problem in polynomial time. This is analogous to what makes a problem NP-hard for the class of problems NP. This means that a polynomial-time algorithm for *any* PPAD-hard problem would immediately yield a polynomial-time algorithm for *every* problem in PPAD. The class of *PPAD-complete* problems is analogous to the class of *NP-complete* problems: the class of PPAD-complete problems is the set of problems that are PPAD hard, while also being contained in PPAD.

To show PPAD hardness of a computational problem class P, one starts from an existing problem class Q known to be PPAD hard (typically Q would be a PPAD-complete problem class, such as computing a Nash equilibrium of an arbitrary two-player zero-sum game). Then one must give two polynomial-time algorithms: First, a polynomial-time algorithm that takes an arbitrary instance from Q and produces an instance of P. Secondly, a polynomial-time algorithm that takes a solution to the constructed instance of P, and constructs a valid solution to the problem from Q based on the solution to P. Such a pair of algorithms immediately implies that any polynomial-time algorithm for the class of problems P would yield a polynomial-time algorithm for the class of problems Q, and thus for the entire class of problems in PPAD due to the PPAD-hardness of Q.

To show PPAD containment of a problem class P, one must go the other way: a polynomial-time algorithm is constructed for reducing an instance of P to an instance of some computational problem Q contained in PPAD, again such that there is a polynomial-time algorithm for constructing a solution to P given a solution to the constructed problem of Q.

The sense in which the Nash equilibrium problem is “hard” is that several computational variants of finding a Nash equilibrium are known to be PPAD-complete problems, and thus a polynomial-time algorithm for finding (approximate) Nash equilibria cannot exist unless the entire class PPAD has polynomial-time algorithms. Let us say that n-NASH is defined as the following problem: we are given an n-player general-sum game and an $\epsilon \geq 0$, and we must find an ϵ-Nash equilibrium of the given game. This problem is known to be hard in the sense that there exists ϵ such that finding an ϵ-Nash equilibrium is a PPAD-complete problem.

Theorem 2.4 *The following problems are PPAD complete:*

- 2-*NASH for* $\epsilon \leq k^{-c}$ *where* k *is the number of actions per player and* $c > 0$.
- n-*NASH for constant* $\epsilon > 0$

In games with more than two players we must allow $\epsilon > 0$ in order to have PPAD completeness. When $\epsilon = 0$, finding an exact Nash equilibrium in $n > 2$ player games is no longer a problem contained in the PPAD class of problems, and thus it is not PPAD complete (though it is still PPAD hard!). Instead, exact n-NASH is contained in the larger class of problems called FIXP, and already for $n = 3$ the problem is complete for the FIXP class. The fundamental reason why exact Nash equilibrium computation cannot be contained in PPAD is that when $n > 2$ there may not even exist a Nash equilibrium described by rational numbers, even if the input to the problem is rational. This precludes containment in PPAD.

A useful way to think of the class PPAD for a non-complexity-theorist is simply as a collection of hard "natural" problems from economics and game theory. A few other notable examples of PPAD-complete problems that are adjacent to problems we will encounter in this book are:

(i) Finding a Fisher market equilibrium (Chapter 11) when each agent has a separable piecewise-linear concave utility function.
(ii) In an Arrow-Debreu exchange economy (Chapter 10), it is PPAD-complete to find an approximate market equilibrium when buyers have Leontief (see Chapter 11) utility functions.
(iii) Computing an approximate *competitive equilibrium from equal incomes* (A-CEEI), a solution concept for fairly allocating indivisible goods such as course seats (Chapter 13).

2.3 Historical Notes

Early pioneers of game theory include Emile Borel and John von Neumann. Perhaps the single most foundational result in the establishment of the field was the proof of von Neumann's minimax theorem in 1928 in his seminal paper (von Neumann, 1928).

The result where Heads-up limit Texas hold'em was *essentially solved* was by Bowling et al. (2015). That paper also introduced the notion of "essentially solved." The strong performance against top-tier humans in 6-player poker was shown by Brown and Sandholm (2019b).

Daskalakis et al. (2009) were the first to show that games beyond two-player zero-sum are PPAD-hard problems. Their initial result was for four-player games. Chen et al. (2009) showed that the result holds even for two-player general-sum games. NP-completeness of finding Nash equilibria with various properties was shown by Gilboa and Zemel (1989) and Conitzer and Sandholm (2008). Codenotti et al. (2006) show that exchange economies with Leontief utilities encode two-player general-sum games, and thus the hardness result of Chen et al. (2009) implies hardness of computing a market equilibrium. Chen and Teng (2009) showed hardness of computing Fisher market equilibrium with separable piecewise-linear concave utilities. Othman et al. (2016) showed the PPAD completeness of finding A-CEEI.

Further reading.

For a classical introduction to game theory, I recommend Osborne and Rubinstein (1994) or Fudenberg and Tirole (1991). These are the standard books used for graduate-level game theory in economics.

For a more technical coverage of the computational complexity of computing equilibria and PPAD problems, I like Roughgarden (2016) as a starting point. There are currently no textbooks covering important recent developments. In the three years prior to the writing of this book, there has been tremendous progress on making it easier to prove both PPAD hardness and PPAD containment. For proving PPAD containment, Filos-Ratsikas et al. (2024) develop a framework based on "convex optimization gates," and show that any problem whose solutions can be expresses in that framework are contained in PPAD. Very loosely speaking, the convex optimization gates allow you to write down a set of convex optimization problems, each of whose input may depend on the output of the other problems. This makes it much simpler to prove PPAD containment for new market equilibrium or game-theoretic equilibrium problems, because such problems can often be phrased as having every player solve a convex optimization problem whose input depends on the output of the other players' optimization problems. It is instructive to think through how one could do this e.g. for the basic Nash equilibrium problem. For proving PPAD hardness, Deligkas et al. (2024) showed that a problem called Pure-Circuit is PPAD-complete. Pure-Circuit is a very attractive starting point for a PPAD hardness reduction: one only needs to show how to encode three or four logical gates in order to show hardness. Moreover, Pure-Circuit is hard to approximate as well, and thus a reduction from Pure-Circuit leads to hardness of approximation as well.

3

Auctions and Mechanism Design Introduction

In this section note we will study the problem of how to aggregate a set of agent preferences into an outcome, ideally in a way that achieves some desirable outcome. Desiderata we might care about include *social welfare*, which is just the sum of the agent's utilities derived from the outcome, or revenue in the context of auctions.

Suppose that we have a car, and we wish to give it to one of n people, with the goal of giving it to the person that would get the most utility out of the car. One thing we could do is ask each person to tell us how much utility they would get out of receiving the car, expressed as some positive number, and then give it to the person that claims to value the car the most. It should be immediately clear that we cannot hope to elicit the true values of the agents this way, since each agent will simply try to name the largest number possible.

This trivial example shows that in general we need to be careful about how we design the rules that map the stated preferences by the agents of a mechanism into an outcome. The general field concerned with the design of such rules is called *mechanism design*. If we are able to charge payments based on the outcome and the reported values then it is possible to achieve a number of attractive properties.

3.1 Auctions

We will mostly focus on the most classical mechanism-design setting: auctions. We will start by considering single-item auctions: there is a single good for sale, and there is a set of n buyers, with each buyer having some value v_i for the good. The goal will be to sell the item via a *sealed-bid* auction, which works as follows:

(i) Each bidder i submits a bid $b_i \geq 0$, without seeing the bids of anyone else.
(ii) The seller decides who gets the good based on the submitted bids.
(iii) Each buyer i is charged a price p_i which is a function of the bid vector b.

A few things in our setup may seem strange. First, most people would not think of sealed bids when envisioning an auction. Instead, they typically envision what's called the *English auction*. In the English auction, bidders repeatedly call out increasing bids, until the bidding stops, at which point the highest bidder wins and pays their last bid. This auction can be conceptualized as having a price that starts at zero, and then rises continuously, with bidders dropping out as they become priced out. Once only one bidder is left, the increasing price stops and the item is sold to the last bidder at that price. This auction format turns out to be equivalent to the *second-price* sealed-bid auction which we will cover below. Another auction format is the *Dutch auction*, which is less prevalent in practice. It starts the price very high such that nobody is interested, and then continuously drops the price until some bidder says they are interested, at which point they win the item at that price. The Dutch auction is likewise equivalent to the *first-price* sealed-bid auction, which we cover below.

Secondly, it would seem natural to always give the item to the highest bid in step 2, but this is not always done (though we will focus on that rule). Thirdly, the pricing step allows us to potentially charge more bidders than only the winner. This is again done in some reasonable auction designs, though we will mostly focus on auction formats where $p_i = 0$ if i does not win.

When thinking about how buyers are going to behave in an auction, we need to first clarify what each buyer knows about the other bidders. Perhaps the most standard setting is one where each buyer i has some distribution F_i from which their value is drawn, independently of the distribution for every other buyer. This is known as the *independent private values* (IPV) model. In this model, every buyer knows the distribution of every other buyer, but they only get to observe their own value $v_i \sim F_i$ before choosing their bid b_i. For this model, we need a new game-theoretic equilibrium notion called a *Bayes Nash equilibrium* (BNE). First, a pure strategy σ_i for a buyer specifies which action they take for every value they may have, so $\sigma_i(v_i)$ is the action taken by buyer i when their own value is v_i. A BNE is then a set of strategies $\{\sigma_i\}_{i=1}^n$, such that for all values v_i and alternative bids b_i, $\sigma_i(v_i)$ achieves at least as much utility as b_i in a Bayesian sense:

$$\mathbb{E}_{v_{-i}\sim F_{-i}}[u_i(\sigma_i(v_i), \sigma_{-i}(v_{-i}))|v_i] \geq \mathbb{E}_{v_{-i}\sim F_{-i}}[u_i(b_i, \sigma_{-i}(v_{-i}))|v_i].$$

In the auction context, $u_i(b_i, \sigma_{-i}(v_{-i}))$ is the utility that buyer i derives given the allocation and payment rule. The idea of a BNE works more generally for a game setup where u_i is some arbitrary utility function.

We will now introduce some useful mechanism-design terminology. We will introduce it in this single-item auction context, but it applies more broadly.

Efficiency. An outcome of a single-item auction is *efficient* if the item ends up allocated to the buyer that values it the most. In general mechanism design problems, an efficient outcome is typically taken to be one that maximizes the sum of the agent utilities, which is also known as maximizing the *social welfare*. Alternatively, efficiency is sometimes taken to mean that we get a Pareto-optimal outcome, which is a weaker notion of efficiency than social welfare maximization (convince yourself of this with a small example.)

Revenue. The revenue of a single-item auction is simply the sum of payments made by the bidders.

Truthfulness, strategyproofness, and incentive compatibility. Informally, we say that an auction is *truthful, strategyproof* or *incentive compatible* (IC) if buyers maximize their utility by bidding their true value (i.e. $b_i = v_i$).[1] More formally, an auction is *dominant strategy incentive compatible* (DSIC) if a buyer maximizes their utility by bidding their value, *no matter what everyone else does*. Saying that an auction (or more generally a mechanism) is "truthful" or "strategyproof" typically means that it is DSIC. We shall adopt that terminology in the book. DSIC auctions are very attractive because buyers do not need to strategize about what the other buyers will do: no matter what happens, they should just bid their value. This also means that, as auction designers, we can reasonably expect that buyers will bid their true value (or at least try to, if they are not perfectly capable of estimating it themselves). This makes it much easier to reason about aspects such as efficiency or revenue.

A slightly weaker degree of incentive compatibility is that of *Bayes-Nash incentive compatibility*: an auction is Bayes-Nash IC if there exists a BNE where every buyer bids their value. It is clear why this notion is less appealing: Now buyers need to worry about whether other buyers are going to bid truthfully. If they think that they will, then we might expect them to bid their value as well. However, if the system starts out in some other state, we might worry that buyers will adapt their bidding over time in a way that pushes them into some other non-truthful equilibrium.

[1] You can tell that game theorists care a lot about truthfulness from the fact that we have at least three names for it!

3.1.1 Second-price auctions

We first look at the *second-price auction*. In a second-price auction, we allocate the item to the highest bid (breaking ties arbitrarily), but the winning bidder i^* is charged the *second-highest bid*. To see why charging the second-highest bid is a good idea, it is helpful to contrast with the *first-price auction* (see the next section). Under the first-price rule, the winner pays their bid. Under this rule, the winning bidder has an incentive to shade their bid such that it is barely above the second-highest bid, because their utility strictly increases as they shade their bid, as long as they still win the item. With the second-price rule, we remove this problem: for the winning bidder, *any* bid higher than the second-highest bid leads to exactly the same outcome for them, and so they do not need to worry about "targeting" the second-highest bid via shading. In fact, it turns out that the second-price auction is truthful because of the above logic.

Theorem 3.1 *The second-price auction is DSIC.*

Proof Consider an arbitrary buyer i with value v_i. Let $b_2 = \max_{k \neq i} b_k$ be the highest bid by any *other* buyer than i. There are four cases to consider for a non-truthful bid $b_i \neq v_i$:

(i) $b_i > v_i \geq b_2$ where b_2 is the second-highest bid. In that case buyer i would have gotten the same utility from bidding their valuation v_i.
(ii) $b_i > b_2 > v_i$ where b_2 is the second-highest bid. In that case buyer i wins, but gets utility $v_i - b_2 < 0$, and they would have been better off bidding their valuation.
(iii) $b_i < b_2 < v_i$ where b_2 is the second-highest bid. In that case buyer i does not win, but they could have won and gotten strictly positive utility if they had bid their valuation.
(iv) $b_2 < b_i < v_i$ where b_2 is the second-highest bid. In that case buyer i wins, but they would have won, and paid the same, if they had bid their true value.

It follows that the second-price auction is DSIC, because an agent should report their true valuation no matter what everybody else does. □

The second-price auction is also efficient, in the sense that it maximizes social welfare (since the item goes to the buyer with the highest value). Finally, it is *computable*, in the sense that it is easy to find the allocation and payments.

3.1.2 First-price auctions

First-price auctions are perhaps what most people imagine when we say that we are selling a good via a sealed-bid auction. In first-price auctions, each buyer submits some bid $b_i \geq 0$, and then we allocate the item to the buyer i^* with the highest bid, and charge that buyer b_{i^*}. This pricing rule is also sometimes referred to as *pay-your-bid*.

Let's briefly try to reason about what might happen in a first-price auction. Clearly, no buyer should bid their true value for the good under this mechanism; in that case they receive no utility even when they win. Instead, buyers should *shade* their bids, so that they sometimes win while also receiving strictly positive utility. The problem is that buyers must strategize about how other buyers will bid, in order to figure out how much to shade by.

This issue of shading and guessing what other buyers will bid happened in early Internet ad auctions, where first-price auctions were initially adopted. *Overture* was an early pioneer in selling Internet sponsored search ads via auction. They initially ran first-price auctions, and provided services to MSN and Yahoo (which were popular search engines at the time). Bidding and pricing turned out to be very inefficient, because buyers were constantly changing their bids in order to best respond to each other. Plots of the price history show a clear "sawtooth pattern," where a pair of bidders will take turns increasing their bid by 1 cent each, in order to beat the other bidder. Finally, one of the bidders reaches their valuation, at which point they drop their bid much lower in order to win something else instead. Then, the winner realizes that they should bid much lower, in order to decrease the price they pay. At that point, the bidder that dropped out starts bidding 1 cent more again, and the pattern repeats. This leads to huge price fluctuations, and inefficient allocations, since about half the time the item goes to the bidder with the lower valuation.

All that said, it turns out that there does exist at least one interesting characterization of how bidding should work in a single-item first-price auction (the Overture example technically consists of many "independent" first-price auctions; though that independence does not truly hold as we shall see later).

For this characterization, we assume the following symmetric model: we have n buyers as before, and buyer i assigns value $v_i \in [0, \overline{v}]$ for the good. Each v_i is sampled i.i.d. from an increasing distribution function F. F is assumed to have a continuous density f and full support. Each bidder knows their own value v_i, but only knows that the value of each other buyer is sampled according to F. Given a bid b_i, buyer i earns utility $v_i - b_i$ if they win, and utility 0 otherwise. If there are multiple bids tied for highest then we assume that a winner is picked uniformly at random among the winning bids, and only the winning bidder pays.

It turns out that there exists a *symmetric equilibrium* in this setting, where each bidder bids according to the function

$$\beta(v_i) = \mathbb{E}[Y_1 | Y_1 < v_i],$$

where Y_1 is the random variable denoting the maximum over $n-1$ independently-drawn values from F.

Theorem 3.2 *If every bidder in a first-price auction bids according to β then the resulting strategy profile is a Bayes-Nash equilibrium.*

Proof Let $G(y) = F(y)^{n-1}$ denote the distribution function for Y_1.

Suppose all bidders except i bid according to β. The function β is continuous and monotonically strictly increasing: a higher value for v_i simply adds additional values to the highest end of the conditional distribution of Y. The highest bid other than that of bidder i is $\beta(Y_1)$. It follows that bidder i should never bid more than $\beta(\overline{v})$, since that is the highest possible other bid. Now consider bidding $b_i \leq \beta(\overline{v})$. By continuity and monotonicity there exists z such that $\beta(z) = b_i$. Notice that $G(z)$ is then the probability of buyer i winning when they bid b_i. The expected value that bidder i obtains from bidding b_i is:

$$\begin{aligned} u_i(b_i, v_i) =& G(z)[v_i - \beta(z)] \\ =& G(z)v_i - G(z)\mathbb{E}[Y_1|Y_1 < z] && \text{by definition of } \beta(z) \\ =& G(z)v_i - \int_0^z yg(y)dy && \text{by definition of expectation} \\ =& G(z)v_i - G(z)z + \int_0^z G(y)dy && \text{integration by parts} \\ =& G(z)(v_i - z) + \int_0^z G(y)dy. \end{aligned}$$

Now we can compare the values from bidding $\beta(v_i)$ and b_i:

$$\begin{aligned} u_i(\beta(v_i), v_i) - u_i(b_i, v_i) =& G(v_i)(v_i - v_i) + \int_0^{v_i} G(y)dy - G(z)(v_i - z) \\ & - \int_0^z G(y)dy \\ =& G(z)(z - v_i) - \int_{v_i}^z G(y)dy. \end{aligned}$$

If $z \geq v_i$ then this is clearly positive since $G(z) \geq G(y)$ for all $y \in [v_i, z]$. If $z \leq v_i$, then $G(z) \leq G(y)$, and so we have a negative number and subtract a more negative number. □

A nice property that follows from the monotonicity of β is that the item is always allocated to the bidder with the highest valuation, and thus the symmetric equilibrium is efficient.

Even though the first-price auction is not truthful, the structure of the equilibrium bidding is quite beautiful. Intuitively, a buyer in the equilibrium could be described as acting according to the following thought process: First, the buyer assumes that they have the highest valuation (if not then they do not want to win because they should be priced out by the higher bidders). Conditional on this information, they calculate the expected value of the second-highest valuation, and submit this as their bid. This price is exactly the price that they would pay in expectation under the second-price auction, conditional on being the highest bid! This is an instantiation of a broader phenomenon known as *revenue equivalence*, which asserts that under symmetric valuations such as the above, all auction formats which always give the item to the buyer with the highest (true) valuation must have equivalent revenue (see the references in the historical notes for books that cover this topic in detail). One immediate consequence is that, if the auctioneer wishes to extract more revenue, then they must give up on maximizing social welfare, and occasionally avoid giving the item to the buyer with the highest valuation (either by withholding the item, or giving it to a lower-ranked buyer).

Like the second-price auction, the first-price auction is computable, and under the symmetric equilibrium given in Theorem 3.2 it is also efficient. But it is not truthful, and it is not hard to come up with a simple discrete setting where there is not even an equilibrium.

3.2 Mechanism Design

More generally, in mechanism design:

- There's a set of outcomes O, and the job of the mechanism is to choose some outcome $o \in O$.
- Each agent i has a private *type* $\theta_i \in \Theta_i$, that they draw from some publicly-known distribution F_i.
- Each agent i has some publicly-known valuation function $v_i(o|\theta_i)$ that specifies a utility for each outcome $o \in O$, given their type $\theta_i \in \Theta_i$.
- The goal of the center is to design a mechanism that maximizes some objective, e.g. social welfare $\sum_i v_i(o|\theta_i)$.

A mechanism takes as input a vector of reported types θ from the players, and outputs an outcome, formally it is a function $f : \times_i \Theta_i \to O$ that specifies the outcome that results from every possible set of reported types. In mechanism design with money, we also have a *payment function* $g : \times_i \Theta_i \to \mathbb{R}^n$ that specifies how much each agent pays given a reported type vector θ.

Let us describe how the general mechanism design setting maps onto the first-price auction setting: the set of outcomes is the n different ways we can allocate the item, and the set of possible reports for buyer i is $\Theta_i = [0, \bar{v}]$, where $\bar{v}$ is some bound on the largest possible bid. A report for a buyer is the bid on the item, which we can think of as their (non-truthful) report of their true value for the item. The valuation function $v_i(o|\theta_i) = \theta_i$ for the outcome o such that buyer i gets the item, and $v_i(o|\theta_i) = 0$ for any outcome o where buyer i does not get the item.

In an ideal mechanism, we have that the mechanism is DSIC, which allows us to analyze outcomes of the mechanism (such as the welfare properties of the mechanism) under the assumption that we have the true type vector. Formally, a mechanism being DSIC would mean that for every agent i, type $\theta_i \in \Theta_i$, any type vector θ_{-i} of the remaining agents, and misreported type $\theta'_i \in \Theta_i$:

$$\mathbb{E}\left[v_i(f(\theta_i, \theta_{-i}))\right] \geq \mathbb{E}\left[v_i(f(\theta'_i, \theta_{-i}))\right],$$

where the expectation is over the potential randomness of the mechanism. If there is also a payment function g and agents have *quasilinear utilities* then the inequality is

$$\mathbb{E}\left[v_i(f(\theta_i, \theta_{-i})) - g(\theta_i, \theta_{-i})\right] \geq \mathbb{E}\left[v_i(f(\theta'_i, \theta_{-i})) - g(\theta'_i, \theta_{-i})\right],$$

Sometimes DSIC is too much to ask for in a given setting. In that case, a weaker form of truthfulness is that there exists a Bayes-Nash equilibrium of the game induced by the mechanism, in which every agent reports their true type. Mechanisms where such an equilibrium exists are called *Bayesian incentive compatible* (usually abbreviated as BIC). Formally, that would mean that for every agent i, type $\theta_i \in \Theta_i$, and misreported type $\theta'_i \in \Theta_i$:

$$\mathbb{E}_{\theta_{-i}}\left[v_i(f(\theta_i, \theta_{-i}))\right] \geq \mathbb{E}_{\theta_{-i}}\left[v_i(f(\theta'_i, \theta_{-i}))\right],$$

where the expectation is over the types θ_{-i} of the other agents, and the potential randomness of the mechanism (note the difference to DSIC, where we did not take expectation over the types of other agents). In words, this constraint just says that reporting an agent's true type should maximize their expected utility, given that everybody else is truthfully reporting. This can likewise be generalized to settings with a payment function g.

In the setting where we can charge money, the *Vickrey-Clarke-Groves* (VCG) mechanism is DSIC and maximizes social welfare. In VCG, after receiving the type vector θ, we pick the outcome o that maximizes the reported welfare.

Formally, VCG selects an outcome o^* in the set $\arg\max_{o \in O} \sum_i v_i(o|\theta_i)$. Of course, an agent i can effectively "choose" the allocation by strategically reporting a type with a high value for a given outcome. The key to making VCG incentive compatible is that we charge each agent their *externality*, which is the amount that their presence in the markets harms the sum of utilities over the remaining agents. Suppose that every agent i' reports a type $\theta'_{i'}$; none of these are assumed to be truthful, since we are trying to show DSIC. In order to define the externality, we discuss the "social welfare" of a given outcome $o \in O$ under the assumption that $\theta'_{i'}$ is the true type of agent $i' \neq i$, since we must use the reported types to measure social welfare in the externality definition. The externality of agent i is then defined as

$$\underbrace{\max_{o' \in O} \sum_{i' \neq i} v_{i'}(o'|\theta'_{i'})}_{\text{optimal welfare without } i} - \underbrace{\sum_{i' \neq i} v_{i'}(o^*|\theta'_{i'})}_{\text{welfare with } i \text{ present}} .$$

The first term is the maximum social welfare achievable when ignoring the utility of agent i (i.e. how well the remaining agents would have done if i left the market), and the second term is the actual sum of utilities achieved by the remaining agents with agent i present. The utility for agent i under a given outcome $o \in O$ is their value for the outcome given their true type θ_i, minus their externality payment:

$$\begin{aligned} & v_i(o|\theta_i) - \underbrace{\max_{o' \in O} \sum_{i' \neq i} v_{i'}(o'|\theta'_{i'}) + \sum_{i' \neq i} v_{i'}(o|\theta'_{i'})}_{\text{externality of } i} \\ &= \underbrace{v_i(o|\theta_i) + \sum_{i' \neq i} v_{i'}(o|\theta'_{i'})}_{\text{social welfare under } o} - \max_{o' \in O} \sum_{i' \neq i} v_{i'}(o'|\theta'_{i'}). \end{aligned}$$

In the second equation we collect all agent values for the outcome, and we see that this is exactly the social welfare under the outcome o. The second term cannot be affected by agent i since their reported type does not factor into the maximization. Thus, the only thing that agent i can do is try to maximize the first term, which is the social welfare measured on θ'_{-i}, but with the true type θ_i for agent i. The first term is maximized by choosing the outcome $o \in O$ that maximizes social welfare, which is achieved by agent i reporting truthfully, since VCG chooses the welfare-maximizing allocation given the reported types.

3.3 Historical Notes

The issues with first-price auctions in the context of Overture's sponsored search auctions are described in Edelman and Ostrovsky (2007), which also shows plots from real data exhibiting the sawtooth pattern. The derivation of the symmetric equilibrium of the first-price auction follows the proof from Krishna (2009). Interestingly, first-price auctions have experienced a resurgence in the internet advertising industry. In the context of display advertising many independent ad exchanges moved to first price in 2018, and Google followed suit and moved their Ad Manager to first price in 2019[2].

The second-price auction is sometimes referred to as the *Vickrey auction* after its inventor (Vickrey, 1961). The generalized second-price auction was described by Edelman et al. (2007), though it had been in use in the Internet ad industry for a while at that point. The VCG mechanism was described in a series of papers by Vickrey (1961), Clarke (1971), and Groves (1973). A full description of a slightly more general VCG mechanism, and proof of correctness, can be found in Nisan et al. (2007, Chapter 9)

Further reading.

As mentioned in the preface, mechanism design is a very deep topic of its own. The reader is encouraged to study the books by Börgers (2015) and Krishna (2009) for a thorough treatment of the topic.

[2] See `https://www.blog.google/products/admanager/update-first-price-auctions-google-ad-manager/`

PART TWO

GAME SOLVING AND REGRET MINIMIZATION

4
Regret Minimization and the Minimax Theorem

So far we have mostly discussed the *existence* of game-theoretic equilibria such as Nash equilibrium. Now we will get started on how to *compute* Nash equilibria, specifically in two-player zero-sum games. The fastest methods for computing large-scale zero-sum Nash equilibrium are based on what's called *regret minimization*.[1] Regret minimization is a form of single-agent decision-making, where the decision maker repeatedly chooses decision from a set of possible choices, and each time they make a decision, they are then given some *loss vector* specifying how much loss they incurred through their decision. It may seem counterintuitive that we move on to a single-agent problem after discussing game-theoretic problems with two or more players, but we shall see that regret minimization can be used to *learn* how to play a game. We will also use it to prove a fairly general version of von Neumann's minimax theorem.

4.1 Regret Minimization

In the simplest regret-minimization setting we imagine that we are faced with the task of repeatedly choosing among a finite set of n actions. At each time step, we choose an action, and then a loss $g_{ti} \in [0, 1]$ is revealed for each action $i \in [n]$. The loss is how *unhappy* we are with having chosen action i, and the goal is to minimize losses over time.[2] This scenario is then repeated iteratively. The losses may be chosen adversarially after we make our choice,

[1] Other popular names that are largely equivalent are *online learning*, *no-regret learning* and *online convex optimization*.

[2] One could equivalently write everything in terms of maximizing rewards instead. This is more natural from a game-theoretic perspective. However, the analysis of these algorithms is heavily rooted in convex minimization theory, and thus it is easier to work with minimization of losses, so that we will not need to translate all the convex minimization machinery into concave maximization machinery.

and we would like to come up with a decision-making procedure that does at least as well as the single best action in hindsight. We will be allowed to choose a distribution over actions, rather than a single action, at each decision point. Classical example applications would be picking stocks repeatedly over time, route selection for driving to work every day, or weather forecasting. To be concrete, imagine that we have n weather-forecasting models that we will use to forecast the weather each day. We would like to decide which model is best to use, but we're not sure how to pick the best one. In that case, we may run a regret-minimization algorithm, where our "action" is to pick a model, or a probability distribution over models, to forecast the weather with. If we spend enough days forecasting, then we will show that it is possible for our *average* prediction to be as good as the best single model in hindsight. As can be seen from the above examples, regret minimization methods are widely applicable beyond equilibrium computation and a useful tool to know about.

4.1.1 Setting

Formally, we are faced with the following problem. At each time step $t = 1, \ldots, T$:

(i) We must choose a decision $x_t \in \Delta^n$
(ii) Afterwards, a loss vector $g_t \in [0, 1]^n$ is revealed to us, and we pay the loss $\langle g_t, x_t \rangle$

Our goal is to develop an algorithm that recommends good decisions. A natural goal would be to do as well as the best sequence of actions in hindsight. But this turns out to be too ambitious, as the following example shows

Example 4.1 We have 2 actions a_1, a_2. At time step t, if our algorithm puts probability greater than $\frac{1}{2}$ on action a_1, then we set the loss to $(1, 0)$, and vice versa we set it to $(0, 1)$ if we put less than $\frac{1}{2}$ on a_1. Now we face a loss of at least $\frac{T}{2}$, whereas the best sequence in hindsight has a loss of 0.

Instead, our goal will be to minimize what is known as *external regret*. The external regret at time t is how much worse our sequence of actions is, compared to the best single action in hindsight:

$$R_t = \sum_{\tau=1}^{t} \langle g_\tau, x_\tau \rangle - \min_{x \in \Delta^n} \sum_{\tau=1}^{t} \langle g_\tau, x \rangle.$$

Because we only work with external regret in this book, we shall simply refer to it as regret going forward.

We say that an algorithm is a *no-regret algorithm* if for every $\epsilon > 0$, there exists a sufficiently-large time horizon T such that the algorithm guarantees $\frac{R_T}{T} \leq \epsilon$ for any input sequence.

Let's see an example showing that randomization over actions is necessary. Consider the following natural algorithm: at time t, choose the action that minimizes the losses seen so far, where e_i is the vector of all zeroes except index i is 1:

$$x_{t+1} = \underset{x \in \{e_1, \ldots, e_n\}}{\arg\min} \sum_{\tau=1}^{t} \langle g_\tau, x \rangle. \tag{FTL}$$

This algorithm is called *follow the leader* (FTL). Note that it always chooses a deterministic action. The following example shows that FTL, as well as any other deterministic algorithm, cannot be a no-regret algorithm.

Example 4.2 At time t, say the algorithm takes action i. Now the adversary can choose the loss vector g_t such that $g_{t,i} = 1$, and $g_{t,j} = 0, \ \forall j \neq i$. Then our deterministic algorithm has loss T at time T, whereas the cost of the best action in hindsight is at most $\frac{T}{n}$ (this follows from the pigeonhole principle). Therefore, the algorithm has linear regret.

It is also possible to derive a lower bound showing that *any* algorithm must have regret at least $O(\sqrt{T})$ in the worst case under adversarial input, see e.g. Roughgarden (2016) Example 17.5. Thus, for adversarial inputs the best we can hope for is a regret guarantee on the order of $O(\sqrt{T})$. If the input sequence is not adversarial then it is possible to do better than $O(\sqrt{T})$. For example, we will see in Chapter 7 that when certain regret minimization algorithms play against each other in a two-player zero-sum game, then they are each guaranteed at most constant regret.

4.1.2 The Multiplicative Weights Algorithm

We now show that, while it is not possible to achieve no-regret with deterministic algorithms, it is possible with ones that play a mixed strategy at every iteration.[3] We will consider the *multiplicative weights update* algorithm.[4] It works as follows:

- At $t = 1$, initialize a weight vector w^1 with $w_i^1 = 1$ for all actions i

[3] In the online learning setting, we assume that a player can choose mixed strategies at every iteration, and that they receive the expected value associated to their chosen mixed strategy. The setting where a player must sample from their distribution, and observes only the loss associated to the played action, is called the *bandit feedback* setting. We do not cover bandit feedback settings in this book.

[4] This algorithm has many names. It is often referred to as *hedge* in the online learning literature.

- At time t, choose actions according to the probability distribution $x_{t,i} = \frac{w_i^t}{\sum_{j\in[n]} w_j^t}$
- After observing g_t, set $w_i^{t+1} = w_i^t \cdot e^{-\eta g_{t,i}}$, where η is a stepsize parameter

The stepsize η controls how aggressively we respond to new information. If $g_{t,i}$ is large then we decrease the weight w_i more aggressively, and thus play action i less frequently.

Theorem 4.3 *Consider running multiplicative weights for T time steps. Then the regret satisfies*

$$R_T \le \frac{\eta T}{2} + \frac{\log n}{\eta}.$$

Proof Let g_t^2 denote the vector of *squared* losses (this square arises from the application of a Taylor expansion below). Let $Z_t = \sum_{j\in[n]} w_j^t$ be the sum of weights at time t. We have

$$\begin{aligned} Z_{t+1} &= \sum_{i=1}^n w_i^t e^{-\eta g_{t,i}} \\ &= Z_t \sum_{i=1}^n x_{t,i} e^{-\eta g_{t,i}} \\ &\le Z_t \sum_{i=1}^n x_{t,i}(1 - \eta g_{t,i} + \frac{\eta^2}{2} g_{t,i}^2) \\ &= Z_t \left(1 - \eta\langle x_t, g_t\rangle + \frac{\eta^2}{2}\langle x_t, g_t^2\rangle\right) \\ &\le Z_t e^{-\eta\langle x_t, g_t\rangle + \frac{\eta^2}{2}\langle x_t, g_t^2\rangle}, \end{aligned}$$

where the first inequality uses the second-order Taylor expansion $e^{-x} \le 1 - x + \frac{x^2}{2}$, which is true for $x \ge 0$, and the second inequality uses $1 + x \le e^x$.

Telescoping and using $Z_1 = n$, we get

$$Z_{T+1} \le n \prod_{t=1}^T e^{-\eta\langle x_t, g_t\rangle + \frac{\eta^2}{2}\langle x_t, g_t^2\rangle} = n e^{-\eta \sum_{t\in[T]}\langle x_t, g_t\rangle + \frac{\eta^2}{2}\sum_{t\in[T]}\langle x_t, g_t^2\rangle}.$$

Now consider the best action in hindsight i^*. We have

$$e^{-\eta \sum_{t\in[T]} g_{t,i^*}} = w_{i^*}^{T+1} \le Z_{T+1} \le n e^{-\eta \sum_{t\in[T]}\langle x_t, g_t\rangle + \frac{\eta^2}{2}\sum_{t\in[T]}\langle x_t, g_t^2\rangle}.$$

Taking logs gives

$$-\eta \sum_{t\in[T]} g_{t,i^*} \leq \log n - \eta \sum_{t\in[T]} \langle x_t, g_t\rangle + \frac{\eta^2}{2} \sum_{t\in[T]} \langle x_t, g_t^2\rangle.$$

Now we rearrange to get

$$R_T \leq \frac{\log n}{\eta} + \frac{\eta}{2} \sum_{t\in[T]} \langle x_t, g_t^2\rangle \leq \frac{\log n}{\eta} + \frac{\eta T}{2},$$

where the last inequality follows from $x_t \in \Delta^n$ and $g_t \in [0,1]^n$. □

If we know T in advance then we can now obtain $O(\sqrt{T})$ regret by choosing η to minimize the right-hand side of Theorem 4.3. In particular, if we set η to be on the order of $\frac{1}{\sqrt{T}}$ then we get that both terms in the bound are $O(\sqrt{T})$.

4.2 Online Convex Optimization

We now generalize the online learning setting from the preceding section. Before, we had n actions and were choosing a probability distribution over them, and receiving a loss associated to each action. In online convex optimization (OCO), we are faced with a similar, but more general, setting. In the OCO setting, we are making decisions from some compact convex set $X \in \mathbb{R}^n$ (analogous to the fact that we were previously choosing probability distributions from Δ^n). After choosing a decision x_t, we suffer a convex loss $f_t(x_t)$. We will assume that f_t is differentiable for convenience, but this assumption is not necessary. As before, we would like to minimize the regret:

$$R_T = \sum_{t\in[T]} f_t(x_t) - \min_{x\in X} \sum_{t\in[T]} f_t(x).$$

We saw in the previous chapter that the follow-the-leader (FTL) algorithm, which always picks the action that minimizes the sum of losses seen so far, does not work. That same argument carries over to the OCO setting. The basic problem with FTL is that it is too unstable: Consider a setting with $X = [-1,1]$ and $f_1(x) = \frac{1}{2}x$. Now suppose that for f_2 and onwards, f_t alternates between $-x$ and x. Then we get that FTL flip-flops between recommending the actions -1 and 1, since they become alternately optimal, and every time the current recommendation ends up being the wrong choice for the next loss.

This motivates the need for a more stable algorithm. What we will do is to smooth out the decision made at each point in time. In order to describe how this smoothing out works we need to take a detour into *distance-generating functions*.

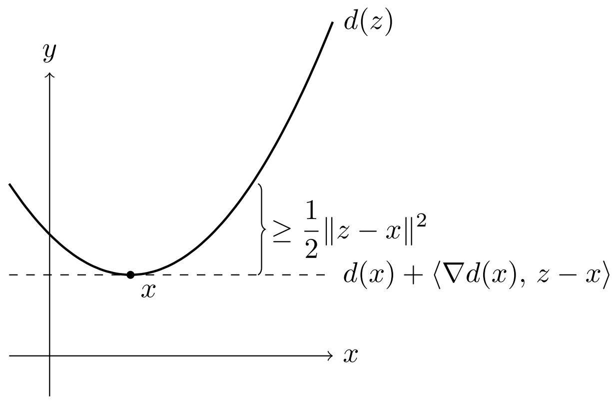

Figure 4.1 Strong convexity illustrated. The gap between the function and its first-order approximation at the point x_0 should grow at least as $\|z - x\|^2$.

4.3 Distance-Generating Functions

A distance-generating function (DGF) is a function $d : X \to \mathbb{R}$ which is differentiable on the relative interior of X, and 1-strongly convex[5] with respect to a given norm $\|\cdot\|$, meaning

$$d(z) \geq d(x) + \langle \nabla d(x), z - x \rangle + \frac{1}{2}\|z - x\|^2, \ \forall x, z \in X.$$

If d is twice differentiable on int X then the following condition is a sufficient, but not necessary, condition for strong convexity with modulus 1 on X:

$$\langle h, \nabla^2 d(x) h \rangle \geq \|h\|^2, \ \forall x \in X, h \in \mathbb{R}^n. \tag{4.1}$$

Intuitively, strong convexity says that the gap between d and its first-order approximation should grow at a rate of at least $\|x - z\|^2$. This property is visualized in Fig. 4.1.

We will use this gap to construct a distance function. In particular, we say that the *Bregman divergence* associated with a DGF d is the function:

$$D(z\|x) = d(z) - d(x) - \langle \nabla d(x), z - x \rangle.$$

Intuitively, we are measuring the distance going from x to z. Note that this is not symmetric, the distance from z to x may be different, and so it is not a true distance metric.

[5] If you have a function f that is α-strongly convex for $\alpha \neq 1$, then f/α is 1-strongly convex.

Given d and our choice of norm $\|\cdot\|$, the performance of our algorithms will depend on the *set width* of X with respect to d:

$$\Omega_d = \max_{x,z\in X} d(x) - d(z),$$

and the dual norm of $\|\cdot\|$:

$$\|g\|_* = \max_{\|x\|\leq 1} \langle g, x\rangle.$$

In particular, our regret bound will require an upper bound on the dual norm of the loss vectors, i.e. $\max_{t\in[T]} \|g_t\|_*$.

Norms and their dual norm satisfy a useful inequality that is often called the generalized Cauchy-Schwarz inequality:

$$\langle g, x\rangle = \|x\| \left\langle g, \frac{x}{\|x\|} \right\rangle \leq \|x\| \max_{\|z\|\leq 1} \langle g, z\rangle \leq \|x\|\|g\|_*.$$

What's the point of these DGFs, norms, and dual norms? The point is that we get to choose all of these in a way that fits the "geometry" of our set X. This will become important once we derive convergence rates that depend on Ω and the dual norm $\|g\|_*$ of the loss vectors.

Consider the following two DGFs for the probability simplex $\Delta^n = \{x : \sum_{i\in[n]} x_i = 1, x \geq 0\}$:

$$d_1(x) = \sum_{i\in[n]} x_i \log(x_i), \quad d_2(x) = \frac{1}{2} \sum_{i\in[n]} x_i^2.$$

The first is the *entropy DGF*,[6] the second is the *Euclidean DGF*. First let us check that they are both strongly convex on Δ^n. The Euclidean DGF is clearly strongly convex with respect to the ℓ_2 norm. The entropy DGF turns out to be strongly-convex with respect to the ℓ_1 norm. The Hessian of the entropy DGF is the diagonal matrix $\nabla^2_{ii} d_1(x) = 1/x_i$. Using the second-order sufficient

[6] The Bregman divergence associated to the entropy DGF is the well-known Kullback-Leibler (KL) divergence.

condition for strong convexity (Eq. (4.1)) and any $h \in \mathbb{R}^n$:

$$\begin{aligned}
\|h\|_1^2 &= \left(\sum_{i\in[n]} |h_i|\right)^2 \\
&= \left(\sum_{i\in[n]} \sqrt{x_i}\frac{|h_i|}{\sqrt{x_i}}\right)^2 \\
&\le \left(\sum_{i\in[n]} x_i\right)\left(\sum_{i\in[n]} \frac{h_i^2}{x_i}\right) && \text{by the Cauchy-Schwarz inequality} \\
&= \left(\sum_{i\in[n]} \frac{h_i^2}{x_i}\right) && \text{because } x \in \Delta^n \\
&= \langle h, \nabla^2 d_1(x) h\rangle.
\end{aligned}$$

Thus, both DGFs have the same strong convexity modulus of one. But now imagine that our losses are in $[0,1]^n$. We denote the maximum dual norm by L. For the Euclidean DGF L is then

$$\max_{\|x\|_2\le 1} \langle \vec{1}, x\rangle = \left\langle \vec{1}, \frac{\vec{1}}{\sqrt{n}}\right\rangle = \sqrt{n}.$$

The set width under the Euclidean DGF is $\Omega_{d_2} = 1$.

In contrast, the maximum dual norm L for the ℓ_1 norm is

$$\max_{\|x\|_1\le 1} \langle \vec{1}, x\rangle = \|\vec{1}\|_\infty = 1,$$

and the set width of the entropy DGF is $\Omega_{d_1} = \log n$.

Now suppose we have a regret bound of the form $O\left(\frac{\Omega L}{\sqrt{T}}\right)$ (we will show such a bound in Theorem 4.6). Then, the entropy DGF gives us a $\log n$ dependence on the number of actions n, whereas the Euclidean DGF leads to a $\sqrt{n}$ dependence. Since n may be very large in some applications, this is a major difference. From the perspective of worst-case regret bounds, the entropy DGF is thus the "right" DGF for the simplex.

In the subsequent analysis, we will need the following inequality on a given norm and its dual norm:

$$\langle g, x\rangle \le \frac{1}{2}\|g\|_*^2 + \frac{1}{2}\|x\|^2. \tag{4.2}$$

which follows from

$$\langle g, x\rangle - \frac{1}{2}\|x\|^2 \le \|g\|_*\|x\| - \frac{1}{2}\|x\|^2 \le \frac{1}{2}\|g\|_*^2,$$

where the first step is by the generalized Cauchy-Schwarz inequality and the second step is by maximizing over x.

We will also need the following result concerning Bregman divergences.

Lemma 4.4 (Three-point lemma) *For any three points x, u, z, we have*

$$D(u\|x) - D(u\|z) - D(z\|x) = \langle \nabla d(z) - \nabla d(x), u - z \rangle.$$

The proof is direct from expanding definitions and canceling terms. The left-hand side is analogous to the triangle inequality. The right-hand side characterizes the difference between the two sides of the "triangle inequality." Unlike the real triangle inequality, here the right-hand side is not guaranteed to be negative. The right-hand side can be seen as an adjustment to the first-order approximation of d at z to u: we subtract out the first-order approximation at x and add in the first-order approximation at z.

4.4 Online Mirror Descent

We now cover one of the canonical OCO algorithms: *Online Mirror Descent* (OMD). In this algorithm, we smooth out the choice of x_{t+1} by penalizing how aggressively we "respond" to $\nabla f_t(x)$ by the Bregman divergence $D(x\|x_t)$ from x_t. This has the effect of stabilizing the algorithm, where the stability is due to the strong convexity of d. We pick our iterates as follows:

$$x_{t+1} = \arg\min_{x \in X} \langle \eta \nabla f_t(x), x \rangle + D(x\|x_t),$$

where $\eta > 0$ is the stepsize.

There is a subtle issue in the above setup. When we measure the Bregman divergence $D(x\|x_t)$, we need to ensure that d is differentiable at x_t. If $x_t \in \operatorname{relint} X$ then this holds by definition, since we defined a DGF as a function that is differentiable on the relative interior. However, it may occur that x_t ends up on the relative boundary of X, in which case the algorithm may be ill-defined unless d is differentiable on the relative boundary. To address this issue, we need to assume that if x_t ends up on the relative boundary, then d is differentiable at that point. One sufficient condition is the following, which ensures that we do not end up on the relative boundary:

$$\lim_{x \to \partial X} \|\nabla d(x)\| = +\infty. \tag{4.3}$$

This condition is satisfied by the entropy DGF. Alternatively, we may simply assume that d is differentiable on all of X (as is the case for the Euclidean DGF).

To ease notation a bit, we let $g_t = \nabla f_t(x_t)$ throughout the section.

We first prove what is sometimes called a *descent lemma* or *fundamental inequality* for OMD.

Theorem 4.5 *For all $x \in X$, we have*

$$\eta(f_t(x_t) - f_t(x)) \le \eta\langle g_t, x_t - x\rangle \le D(x\|x_t) - D(x\|x_{t+1}) + \frac{\eta^2}{2}\|g_t\|_*^2.$$

Proof The first inequality in the theorem is direct from convexity of f_t. Thus, we only need to prove the second inequality.

By first-order optimality of x_{t+1} we have

$$\langle \eta g_t + \nabla d(x_{t+1}) - \nabla d(x_t), x - x_{t+1}\rangle \ge 0, \forall x \in X \tag{4.4}$$

Now pick some arbitrary $x \in X$. By rearranging terms and adding and subtracting $\langle \nabla d(x_{t+1}) - \nabla d(x_t), x - x_{t+1}\rangle$ we have

$$\begin{aligned}
\langle \eta g_t, x_t - x\rangle =& \langle \nabla d(x_t) - \nabla d(x_{t+1}) - \eta g_t, x - x_{t+1}\rangle + \langle \nabla d(x_{t+1}) - \nabla d(x_t), x - x_{t+1}\rangle \\
&+ \langle \eta g_t, x_t - x_{t+1}\rangle \\
\le& \langle \nabla d(x_{t+1}) - \nabla d(x_t), x - x_{t+1}\rangle + \langle \eta g_t, x_t - x_{t+1}\rangle. \\
=& D(x\|x_t) - D(x\|x_{t+1}) - D(x_{t+1}\|x_t) + \langle \eta g_t, x_t - x_{t+1}\rangle \\
\le& D(x\|x_t) - D(x\|x_{t+1}) - D(x_{t+1}\|x_t) + \frac{\eta^2}{2}\|g_t\|_*^2 + \frac{1}{2}\|x_t - x_{t+1}\|^2 \\
\le& D(x\|x_t) - D(x\|x_{t+1}) + \frac{\eta^2}{2}\|g_t\|_*^2.
\end{aligned}$$

The first inequality is by Eq. (4.4). The second equality is by the three-points lemma. The second inequality is by Eq. (4.2). The last inequality is by strong convexity of d. This proves the theorem. □

The descent lemma gives us a one-step upper bound on how much better x is than x_t. Based on the descent lemma, a bound on the regret of OMD can be derived. The idea is to apply the descent lemma at each time step, and then showing that when we sum across the resulting inequalities, a sequence of useful cancellations occur.

Theorem 4.6 *The OMD algorithm with DGF d achieves the following bound on regret:*

$$R_T \le \frac{D(x\|x_1)}{\eta} + \frac{\eta}{2}\sum_{t\in[T]}\|g_t\|_*^2.$$

Proof Consider any $x \in X$. Now we apply the inequality from Theorem 4.5 separately to each time step $t = 1, \ldots, T$, divide through by η, and then summing from $t = 1, \ldots, T$ we get

$$\begin{aligned}\sum_{t\in[T]} \langle g_t, x - x_t \rangle &\le \sum_{t\in[T]} \frac{1}{\eta}\left(D(x\|x_t) - D(x\|x_{t+1}) + \frac{\eta^2}{2}\|g_t\|_*^2 \right) \\ &\le \frac{D(x\|x_1) - D(x\|x_{T+1})}{\eta} + \sum_{t\in[T]} \frac{\eta}{2}\|g_t\|_*^2 \\ &\le \frac{D(x\|x_1)}{\eta} + \sum_{t\in[T]} \frac{\eta}{2}\|g_t\|_*^2,\end{aligned}$$

where the second inequality is by noting that the term $D(x\|x_t)$ appears with a positive sign at the t'th part of the sum, and negative sign at the $t-1$'th part of the sum. □

Notice that in Theorem 4.6, we did not use the boundedness of X. Boundedness of X is only used to ensure that $D(x\|x_1) \le \Omega$ is finite. However, Theorem 4.6 applies more broadly, and we shall use it in an unbounded setting later in Chapter 17.

Suppose that each f_t is Lipschitz in the sense that $\|g_t\|_* \le L$. Using our bound Ω on DGF differences, and supposing we initialize x_1 at the minimizer of d, then we can set $\eta = \frac{\sqrt{2\Omega}}{L\sqrt{T}}$ to get

$$R_T \le \frac{\Omega}{\eta} + \frac{\eta T L^2}{2} \le \sqrt{2\Omega T} L.$$

A related algorithm is the *follow-the-regularized-leader* algorithm. It works as follows:

$$x_{t+1} = \arg\min_{x\in X} \eta \sum_{\tau\in[t]} \langle g_t, x \rangle + d(x).$$

Note that it is more directly related to FTL: it uses the FTL update, but with a single smoothing term $d(x)$, whereas OMD re-centers a Bregman divergence at $D(\cdot\|x_t)$ at every iteration. FTRL can be analyzed similarly to OMD. It gives the same theoretical properties for our purposes, but we will see some experimental performance from both algorithms later where the performance differs quite a bit. For a convergence proof see Orabona (2019, chapter 7).

4.5 Minimax theorems via OCO

In the first and second chapters we saw von Neumann's minimax theorem, which was:

Theorem 4.7 (von Neumann's minimax theorem) *Every two-player zero-sum game has a unique value v, called the* value of the game*, such that*

$$\min_{x \in \Delta^n} \max_{y \in \Delta^m} \langle x, Ay \rangle = \max_{y \in \Delta^m} \min_{x \in \Delta^n} \langle x, Ay \rangle = v.$$

We will now prove a generalization of this theorem.

Theorem 4.8 (Generalized minimax theorem) *Let $X \in \mathbb{R}^n, Y \in \mathbb{R}^m$ be compact convex sets. Let $f(x, y)$ be continuous, convex in x for a fixed y, and concave in y for a fixed x. Assume that f has bounded partial subgradients, i.e. $\partial_x \|f(x, y)\|_2 \le L$, $\partial_y \|f(x, y)\|_2 \le L$ for all $x \in X, y \in Y$. Then there exists a value v such that*

$$\min_{x \in X} \max_{y \in Y} f(x, y) = \max_{y \in Y} \min_{x \in X} f(x, y) = v.$$

Proof We will view this as a game between a player choosing the minimizer x and a player choosing the maximizer y. Suppose first that the maximizer has to go first, and let y^* be an optimal solution to the max-min problem. Now suppose that the maximizer goes second. When y is chosen second the maximizer can observe the chosen x before making a decision. In the worst case, they can always pick y^*, but they may do better. Thus, we get

$$\max_{y \in Y} \min_{x \in X} f(x, y) \le \min_{x \in X} \max_{y \in Y} f(x, y).$$

In order to prove the other direction we will use our OCO results. We run a repeated game where the players choose a strategy x_t, y_t at each iteration t. The x player chooses x_t according to a no-regret algorithm (say OMD), while y_t is always chosen as $\arg\max_{y \in Y} f(x_t, y)$. Let the average strategies be

$$\bar{x} = \frac{1}{T} \sum_{t \in [T]} x_t, \quad \bar{y} = \frac{1}{T} \sum_{t \in [T]} y_t.$$

Using OMD with the Euclidean DGF (since X is compact this is well-defined), we get the following bound:

$$R_T = \sum_{t \in [T]} f(x_t, y_t) - \min_{x \in X} \sum_{t \in [T]} f(x, y_t) \le O\left(\sqrt{\Omega T} L\right). \tag{4.5}$$

Now we bound the value of the min-max problem as

$$\min_{x\in X}\max_{y\in Y} f(x,y) \le \max_{y\in Y} f(\bar{x},y) \le \frac{1}{T}\max_{y\in Y}\sum_{t\in[T]} f(x_t,y) \le \frac{1}{T}\sum_{t\in[T]} f(x_t,y_t),$$

where the first inequality follows because $\bar{x}$ is a valid choice in the minimization over X, the second inequality follows by convexity, and the third inequality follows because y_t is chosen to maximize $f(x_t, y_t)$. Now we can use the regret bound (4.5) for OMD to get

$$\begin{aligned}\min_{x\in X}\max_{y\in Y} f(x,y) &\le \frac{1}{T}\min_{x\in X}\sum_{t\in[T]} f(x,y_t) + O\left(\frac{\sqrt{\Omega}L}{\sqrt{T}}\right)\\ &\le \min_{x\in X} f(x,\bar{y}) + O\left(\frac{\sqrt{\Omega}L}{\sqrt{T}}\right)\\ &\le \max_{y\in Y}\min_{x\in X} f(x,y) + O\left(\frac{\sqrt{\Omega}L}{\sqrt{T}}\right),\end{aligned}$$

where the second inequality is by concavity. Taking the limit as $T \to \infty$ we get

$$\min_{x\in X}\max_{y\in Y} f(x,y) \le \max_{y\in Y}\min_{x\in X} f(x,y),$$

which concludes the proof. □

For simplicity, we assumed continuity of f. The argument did not really need continuity, though. The same proof continues to work for f which is lower and upper semicontinuous in x and y, respectively.

4.6 Historical notes

When applied to the offline setting where $f_t = f\ \forall t$, OMD is equivalent to the *mirror descent* algorithm which was introduced by Nemirovsky and Yudin (1983), with the more modern variant introduced by Beck and Teboulle (2003). There's a functional-analytic interpretation of OMD and mirror descent where one views d as a *mirror map* that allows us to think of f and x in terms of the dual space of linear forms. This was the original motivation for mirror descent, and allows one to apply the algorithm in broader settings, e.g. Banach spaces. This is described in several textbooks and lecture notes e.g. Orabona (2019) or Bubeck *et al.* (2015). The FTRL algorithm run on an offline setting with $f_t = f$ becomes equivalent to Nesterov's *dual averaging* algorithm (Nesterov, 2009).

The minimax theorems in Theorem 4.7 and Theorem 4.8 were developed by John von Neumann in (von Neumann, 1928). The term "von Neumann's minimax theorem" is often used to refer to the specific version in Theorem 4.7. In his original 1928 paper, von Neumann actually proved a more general result for continuous quasi-convex-quasi-concave functions f, which captures the form given in Theorem 4.8. See Kjeldsen (2001) for a discussion of the history of von Neumann's development and conceptualization of the minimax theorem, including a discussion of the quasi-convex-quasi-concave generalization. The more general Theorem 4.8, as well as even more general versions that allow quasi-concavity and quasi-convexity and abstract topological decision spaces, are often referred to as *Sion's minimax theorem*[7], sometimes even in cases that fall under von Neumann's generalization beyond the bilinear case. For example, in his 1958 paper (Sion *et al.*, 1958), Sion claims that von Neumann's theorem is only concerned with bilinear functions, whereas it is actually substantially more general. This misconception that von Neumann only dealt with the bilinear case may have arisen because that is by far the most important case from a game-theoretic perspective (since it enables solutions to two-player zero-sum games). Moreover, von Neumann's original 1928 paper was written in German, and an English translation did not appear until 1958 (von Neumann, 1959).

Further reading.

A very broad coverage of online convex optimization can be found in Orabona (2019). I suggest starting with this book, though some of the below books might be more approachable depending on the reader's background. For a more approachable first text, Hazan *et al.* (2016), which is a very readable introduction to OCO and regret minimization. Another good earlier book is Bubeck *et al.* (2015). Beck (2017) is an excellent reference for a convex optimization perspective on first-order methods.

[7] A quite general version of what's usually referred to as Sion's minimax theorem can be found on Wikipedia at `https://en.wikipedia.org/wiki/Sion%27s_minimax_theorem`.

5

Blackwell Approachability and Regret Matching

In this chapter we are going to introduce a new type of online-learning problem concerned with *vector-valued games*. This framework will eventually be shown to lead to one of the fastest algorithms for game solving in practice.

5.1 Blackwell Approachability

In two-player zero-sum games we saw that there exists a value for the game v such that the row player can choose a strategy x assuring that the payoff will be in the set $(-\infty, v]$ no matter what the column player does. Conversely, the column player can assure that the payoff lies in the set $[v, \infty)$, no matter what the row player does.

In Blackwell approachability we ask whether there is a way to generalize the notion of forcing the payoffs to lie in a particular set to *vector-valued games*.

We consider the following setup:

- Players 1 and 2 choose strategies from compact convex sets X and Y respectively.
- There is a bilinear vector-valued payoff function $f(x, y) \in \mathbb{R}^m$.
- There is a closed convex *target set* C.
- We will assume that $f(x, y) \in B(0, 1)$, where $B(0, 1) = \{g : \|g\|_2 \leq 1\}$.

Player 1 wants to force the payoffs $f(x, y)$ to lie inside C. The case of a single-shot game is trivially analyzed: it is generally only possible to do this if there exists x such that $f(x, y) \in C$, $\forall y \in Y$. So in general this won't be possible. However, it turns out that in a repeated game setting, there is a sense in which the x player can have the payoffs *approach* C. In the repeated game, the players choose actions x_t, y_t at each time step t. The goal for player 1 is to have the average payoff vector $\bar{f}_t = \frac{1}{t} \sum_{i=1}^{t} f(x_t, y_t)$ approach C, while the goal of player 2 is to keep $\bar{f}_t$ from approaching C. We will measure the distance as $d(\bar{f}_t, C) = \min_{z \in C} \|z - \bar{f}_t\|_2$. Formally, we say the following.

Definition 5.1 A target set C is *approachable* if there exists an algorithm for picking x_t based on $x_1, \ldots, x_{t-1}, y_1, \ldots, y_{t-1}$ such that $d(\bar{f}_t, C) \to 0$ as t goes to infinity.

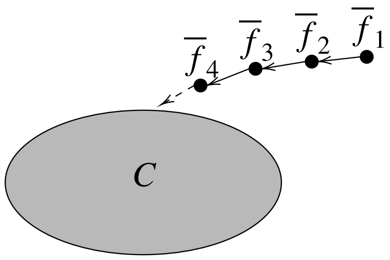

Figure 5.1 Blackwell approachability requires that the sequence $\{\bar{f}_t\}_{t=1}$ approaches C no matter the choices of the y player.

A stronger notion is that player 1 can force a given set H. We already concluded that this will not be possible for the target set other than in trivial cases, but nonetheless, the idea of forcing sets will play a crucial role, particularly forcing hyperplanes that contain the target set C.

Definition 5.2 A set H is *forceable* if there exists x such that $f(x, y) \in C$ for all $y \in Y$.

5.1.1 Scalar Approachability

In the special case where $m = 1$ we get a scalar approachability game. As discussed at the beginning of this section, this can be analyzed via minimax theorems. In particular, for the scalar case target sets are intervals, and we may analyze only intervals of the form $(-\infty, \lambda]$ without loss of generality. Clearly, an interval $(-\infty, \lambda]$ is approachable if $\lambda \geq v$, where v is the value of the game associated to the bilinear function f in Sion's minimax theorem. This follows because if player 1 plays any strategy x such that they are guaranteed at least v, then $f(x, y_t) \in (-\infty, \lambda]$ for all t no matter the y_t. Conversely, if $\lambda < v$, then by Sion's theorem player 2 may play a strategy y such that no matter the x_t, $f(x_t, y) \geq v > \lambda$. We thus have the lemma

Lemma 5.3 *In scalar approachability games, the following three statements are equivalent:*

- *A target set* $(-\infty, \lambda]$ *is approachable.*
- *A target set* $(-\infty, \lambda]$ *is forceable.*
- $\lambda \geq v$, *where* v *is the value of the game associated to* f, X, Y *in Sion's minimax theorem.*

Thus, in the scalar case of Blackwell approachability, approachability of any target set C boils down to whether C intersects with the halfspace (i.e. interval) $(-\infty, \lambda)$. This continues to hold for halfspaces in higher dimension, but non-halfspace target sets become more nuanced.

5.1.2 Halfspace Approachability

We first analyze the special case where the target set is a halfspace $H = \{h : \langle h, a \rangle \leq b\}$. Halfspaces turn out to have the nice property that forceability is equivalent to approachability:

Lemma 5.4 *A halfspace H is approachable if and only if it is forceable.*

Proof The proof consists in reducing halfspace approachability to a scalar approachability game. To do that, let $\hat{f}(x, y) = \langle a, f(x, y) \rangle$. Now we clearly have that forcing H wrt. f is equivalent to forcing $(-\infty, b]$ wrt. $\hat{f}$. Say x^* forces $(-\infty, b]$, then

$$b \geq \hat{f}(x^*, y) = \langle a, f(x^*, y) \rangle, \ \forall y \in Y,$$

and so x^* also forces H, and vice versa.

For approachability, we have that the distance from $\bar{f}_t$ to H satisfies

$$d(\bar{f}_t, H) = d\left(\langle a, \bar{f}_t \rangle, (-\infty, b]\right) = d\left(\frac{1}{t}\sum_{i=1}^{t}\langle a, f_i \rangle, (-\infty, b]\right).$$

Thus, approachability of H is equivalent to approachability of $(-\infty, b]$.

From Lemma 5.3 we have that approachability and forceability are equivalent for $(-\infty, b]$, so they must be equivalent for H. □

5.1.3 Blackwell's Approachability Theorem

Now we are ready to analyze the general case of when a convex closed set C is approachable. Blackwell proved the following:

Theorem 5.5 *A convex closed set C is approachable if and only if every halfspace $H \supseteq C$ is forceable. Moreover, if every halfspace is forceable then there exists a procedure such that the distance between C and the average payoff vector is bounded by $\frac{2}{\sqrt{T}}$ at time T.*

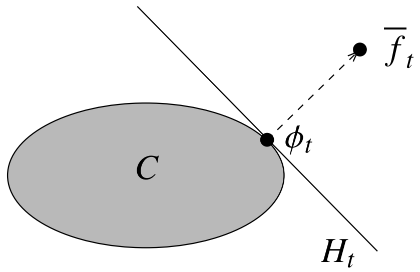

Figure 5.2 The tangent halfspace forced in Blackwell's theorem.

Blackwell's proof is constructive. It is based on the following algorithm for approaching C when all halfspaces containing C are forceable: At every time step t, do the following:

- If $\bar{f}_t \in C$, play any x_t.
- Else consider the projection ϕ_t of $\bar{f}_t$ onto C. We construct a halfspace H_t with normal vector $a_t = \phi_t - \bar{f}_t$, and constant $b_t = \langle a_t, \phi_t \rangle$. Play any x_t forcing H_t. Fig. 5.2 illustrates how H_t is chosen.

The algorithm repeatedly takes the halfspace tangent to the projection of $\bar{f}_t$, and forces it. We now prove Blackwell's theorem.

Proof Say that C is approachable. Then we may play any algorithm guaranteed to approach C, and we will then be guaranteed to approach every $H \supseteq C$.

Now assume that all $H \supseteq C$ are approachable, and play Blackwell's algorithm. First note that since ϕ_t is the projection of $\bar{f}_t$ onto a convex set H_t (this follows from how we constructed H_t) we have from first-order optimality:

$$\langle \phi_t - \bar{f}_t, z - \phi_t \rangle \geq 0, \ \forall z \in H_t. \tag{5.1}$$

Let $f_{t+1} = f(x_{t+1}, y_{t+1})$. We have

$$
\begin{aligned}
d(\bar{f}_{t+1}, C)^2 =& \min_{z \in C} \|\bar{f}_{t+1} - z\|_2^2 \\
\leq& \|\bar{f}_{t+1} - \phi_t\|_2^2 \\
=& \left\| \frac{t}{t+1}\bar{f}_t + \frac{1}{t+1} f_{t+1} - \phi_t \right\|_2^2 ; \quad \text{by definition of } \bar{f}_{t+1} \\
=& \left\| \frac{t}{t+1}(\bar{f}_t - \phi_t) + \frac{1}{t+1}(f_{t+1} - \phi_t) \right\|_2^2 \\
=& \frac{1}{(t+1)^2}\left(t^2 \|(\bar{f}_t - \phi_t)\|_2^2 + \|(f_{t+1} - \phi_t)\|_2^2 + 2t\langle \bar{f}_t - \phi_t, f_{t+1} - \phi_t \rangle \right) \\
\leq& \frac{1}{(t+1)^2}\left(t^2 \|(\bar{f}_t - \phi_t)\|_2^2 + \|(f_{t+1} - \phi_t)\|_2^2 \right); \quad \text{by (5.1)} \\
=& \frac{1}{(t+1)^2}\left(t^2 d(\bar{f}_t, C)^2 + \|(f_{t+1} - \phi_t)\|_2^2 \right).
\end{aligned}
$$

Telescoping this inequality we have

$$
d(\bar{f}_{t+1}, C)^2 \leq \frac{1}{(t+1)^2} \sum_{i=1}^{t} \|f_{t+1} - \phi_t\|_2^2 \leq \frac{4t}{(t+1)^2} \leq \frac{4}{t+1},
$$

where the second inequality is from the fact that we assumed payoffs lie in the norm-ball $B(0,1)$. Taking the square root of both sides gives the theorem. □

5.2 Regret Matching

Blackwell's constructive result can be used to develop regret-minimization algorithms for a variety of sets. A well-known algorithm that we will study is *regret matching*, which is an instantiation of Blackwell's result for regret minimization over the simplex Δ^n with linear losses $g_t \in [0,1]^n$. The instantiation works as follows. We let our decision set be $X = \Delta^n$, and our target set is the nonpositive orthant $C = \mathbb{R}^n_{\leq 0}$. For each pure action i we say that $r_{t,i} = \langle g_t, x_t \rangle - g_{t,i}$ is the regret from not playing action i rather than x_t at time $t \in [T]$, and we let r_t be the vector of all n regrets. We will use $\frac{r_t}{\sqrt{n}}$ as our vector-valued payoff at time t in the Blackwell approachability problem. Note that the regret is now $R_T = \max_i \sum_{t \in [T]} r_{t,i}$, and having regret grow sublinearly is equivalent to $\bar{r}_t = \frac{1}{t}\sum_{k=1}^{t} r_k$ approaching the nonpositive orthant $\mathbb{R}^n_{\leq 0}$, i.e. our target set.

Proposition 5.6 *Suppose we have a regret minimization problem over the simplex Δ^n with linear losses $g_t \in [0,1]^n$, and we use Blackwell's algorithm* *on the approachability instance above. Then the regret is upper bounded as* $R_T \le \sqrt{nT}$.

Proof We have

$$\begin{aligned} R_T &= \max_{i\in[n]} \sum_{t\in[T]} \langle g_t, x_t - e_i\rangle \\ &= \max_{i\in[n]} \sum_{t\in[T]} r_{t,i} \\ &= T \max_{i\in[n]} \bar{r}_{T,i} \\ &\le T \max_{i\in[n]} [\bar{r}_{T,i}]^+ \\ &\le T\sqrt{n}\|[\bar{r}_{T,i}]^+/\sqrt{n}\|_2 \\ &\le \sqrt{Tn}. \end{aligned}$$

The first inequality is simply by thresholding values at zero, the second inequality is by norm equivalence (i.e. $\|\cdot\|_\infty \le \|\cdot\|_2$) and homogeneity of norms, and the third inequality is by Theorem 5.5 after noting that the normalized payoff vector satisfies $\|r_t/\sqrt{n}\|_2 \le 1$, and noting that the Euclidean distance to the positive orthant is exactly the norm of the positively-thresholded vector. □

By Blackwell's theorem having $\bar{r}_t$ approach $\mathbb{R}^n_{\le 0}$ can be done by repeatedly forcing tangent halfspaces. To do so, let ϕ_t be the projection of $\bar{r}_t$ onto $\mathbb{R}^n_{\le 0}$. Note that the normal vector $a_t = \bar{r}_t - \phi_t$ simply thresholds $\bar{r}_t$ at zero, setting all negative entries to zero. Now, we will force r_{t+1} to be in the halfspace with normal vector a_t by ensuring $\langle a_t, r_{t+1}\rangle = 0$. To do so, first consider the square matrix of pairwise regrets B, where B_{ij} is the regret incurred by playing j rather than i under g_{t+1}. We have that $B_{ij} = -B_{ji}$, so B is skew-symmetric, which means that $\langle q, Bq\rangle = 0$ for all q. We set $x_{t+1} = \frac{a_t}{\|a_t\|_1}$, in which case we get that the next regret is $r_{t+1} = Bx_{t+1} = B\frac{a_t}{\|a_t\|_1}$, and now it satisfies $\langle a_t, r_{t+1}\rangle = 0$, and thus we forced the desired halfspace.

Summarizing what we did in terms of our standard regret minimization setup, we have an algorithm that works as follows:

- Play arbitrary x_1.
- Keep a sum $\hat{r}_t = \sum_{k=1}^t r_k$ of regret vectors.
- At time $t+1$ set $x_{t+1,i} = \frac{[\hat{r}_{t,i}]^+}{\sum_{k=1}^n [\hat{r}_{t,k}]^+}$.
- If no regrets are positive, play uniform strategy.

This algorithm is called *regret matching*. By Theorem 5.5 regret matching has

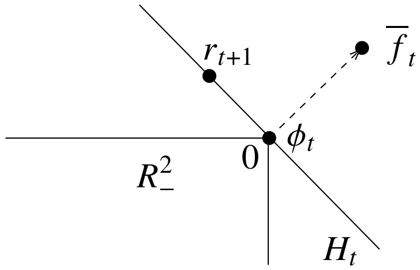

Figure 5.3 The next regret vector r_{t+1} lies in the halfspace forced in Blackwell's theorem.

regret that grows on the order of $O\left(\sqrt{T}\right)$, assuming $g_t \in B(0,1)$ for all t (if this does not hold we may simply normalize the payoffs).

5.3 Regret Matching$^+$

Finally, we present a variation on regret matching, which turns out to be immensely useful in practice. In regret matching, remember that we took the sum of the regret vectors and thresholded it at zero when generating x_{t+1}. In *regret matching*$^+$ (RM$^+$), we only keep track of positive regrets. Formally, we have the following algorithm:

- Initialize $Q_1 = 0$ and play x_1 arbitrarily.
- After seeing r_t, set $Q_t = \left[\frac{t-1}{t} Q_{t-1} + \frac{1}{t} r_t\right]^+$.
- At time $t+1$, play $x_{t+1,i} = \frac{Q_{t,i}}{\|Q_t\|_1}$.

The important observation for RM$^+$ is that we are constructing a sequence that upper-bounds regret, i.e. $Q_t \geq \bar{r}_t$. This is easy to see, as we are only dropping negative terms in the summation that makes up $\bar{r}_t$.

Visually, we may think of it as moving along a face of $\mathbb{R}^n_{\leq 0}$, while maintaining the same distance d to $\mathbb{R}^n_{\leq 0}$ while moving towards 0. See Figure 5.4.

Theorem 5.7 *Assume that the payoff vectors satisfy $g_t \in B(0,1)$. RM$^+$ approaches $C = \mathbb{R}^n_{\leq 0}$ at a rate of $\frac{2}{\sqrt{T+1}}$.*

Proof Let $Q^*_t = 0$ be the projection of Q_t onto C. Let H be the halfspace $\{q : \langle Q_t, q\rangle \leq 0\}$ corresponding to forcing in Blackwell's theorem (since $Q^*_t =$

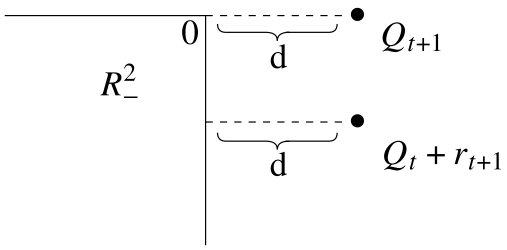

Figure 5.4 The thresholding used in constructing Q_{t+1}.

0). We have

$$
\begin{aligned}
d(Q_{t+1}, C)^2 &= \min_{z \in C} \|Q_{t+1} - z\|^2 \\
&\leq ||Q_{t+1} - Q_t^*||^2 \\
&= \|Q_{t+1}\|^2; \text{ since } Q_t^* = 0 \\
&= \left\| \left[\frac{t}{t+1} Q_t + \frac{1}{t+1} r_{t+1} \right]^+ \right\|^2 \\
&\leq \left\| \frac{t}{t+1} Q_t + \frac{1}{t+1} r_{t+1} \right\|^2 ; \text{ since thresholding can only decrease the norm} \\
&= \frac{1}{(t+1)^2} \left(t^2 \|Q_t\|^2 + \|r_{t+1}\|^2 + 2t \langle Q_t, r_{t+1} \rangle \right) \\
&= \frac{1}{(t+1)^2} \left(t^2 \|Q_t\|^2 + \|r_{t+1}\|^2 \right) ; \text{ by forcing } r_{t+1} \in H.
\end{aligned}
$$

By telescoping we now get

$$
\begin{aligned}
d(Q_{t+1}, C)^2 &\leq \frac{1}{(t+1)^2} \left(t^2 d(Q_t, C) + \|r_{t+1}\|^2 \right) \\
&\leq \frac{1}{(t+1)^2} \sum_{k=1}^{t} \|r_{k+1}\|^2 \\
&\leq \frac{1}{(t+1)^2} 4t \\
&\leq \frac{4}{(t+1)}.
\end{aligned}
$$

Taking square roots concludes the theorem. □

Using the same logic as in Proposition 5.6, we get that RM^+ has a regret bound of $\sqrt{nT}$.

5.4 Overview of Regret Minimizers

At this point we have covered quite a few regret minimizers. In the coming chapters we will start to look at how they can be used to solve zero-sum games, both matrix games and extensive-form games. For now, let us quickly recap and compare our options. Say that we want to minimize linear losses from $[0, 1]^n$ over a simplex Δ^n (note that this covers convex losses with bounded dual norm of the gradients). In that case we have covered 5 algorithms with two different types of regret bounds:

- Regret bound: $O\left(\sqrt{T \log n}\right)$: Hedge and OMD with the entropy DGF (in fact, these are two different perspectives on the same algorithm).
- Regret bound: $O\left(\sqrt{nT}\right)$: OMD (Euclidean), Regret Matching, and Regret Matching$^+$.

It is clear that the entropy-based approach leads to a much more desirable dependence on the dimension of the problem. However, once we start solving games using regret minimization in Chapter 6 we will see that the numerical performance is inverted: the best methods are based on the Euclidean DGF and regret matching$^+$.

5.5 Historical Notes

Blackwell approachability was introduced in Blackwell (1956). Regret matching was introduced by Hart and Mas-Colell (2000). The RM$^+$ algorithm was introduced in Tammelin (2014) and its $O(\sqrt{T})$ regret bound was proven by Tammelin et al. (2015), though not through a Blackwell approachability perspective. The proof of RM$^+$ via modified Blackwell approachability is, I believe, new. It was developed together with Gabriele Farina when working on the papers Farina et al. (2017, 2019b), though we never used it in those works. There exist extensions of the regret matching reduction to other convex compact sets based on Blackwell approachability. See Abernethy et al. (2011) for a general reduction, and Grand-Clément and Kroer (2024) for a numerically-performant procedure for solving more general convex-concave saddle-point problems using Blackwell approachability.

Further reading.

Unfortunately there aren't many places to find coverage of Blackwell approachability, and furthermore all the sources I know of cover it in quite different ways and levels of generality. Lecture notes 13 and 14 of Ramesh Johari (Johari, 2007) cover the finite-action space case as well as regret matching and the relationship to calibration. Another nice presentation for that same case is the one given by Young (2004). The more general proof of Blackwell's theorem given here largely follows the one given in a blog post by Farina at `http://www.cs.cmu.edu/~gfarina/2016/approachability/`. The recently-updated edition of Hazan *et al.* (2016) also added a chapter on Blackwell approachability.

6
Self-Play via Regret Minimization

We have covered a slew of no-regret algorithms: hedge, online mirror descent (OMD), regret matching (RM), and RM$^+$. All of these algorithms can be used for the case of solving two-player zero-sum matrix games of the form

$$\min_{x \in \Delta^n} \max_{y \in \Delta^m} \langle x, Ay \rangle.$$

Matrix games are a special case of the more general saddle-point problems

$$\min_{x \in X} \max_{y \in Y} f(x, y),$$

where f is convex-concave, meaning that $f(\cdot, y)$ is convex for all fixed y, $f(x, \cdot)$ is concave for all fixed x. In this chapter we will cover how to solve this more general class of saddle-point problems by using regret minimization for each "player" and having the regret minimizers perform what is usually called *self play*. The name self play comes from the fact that we usually use the same regret-minimization algorithm for each player, and so in a sense this approach towards computing equilibria lets the chosen regret-minimization algorithm play against itself. After covering the self play setup, we will look at some experiments on practical performance for the matrix-game case.

6.1 From Regret to Nash Equilibrium

In order to use regret-minimization algorithms for computing Nash equilibrium, we will run a repeated game between the x and y players. We will assume that the players have access to regret-minimizing algorithms $\mathcal{A}_x$ and $\mathcal{A}_y$ (we will be a bit loose with notation here and implicitly assume that $\mathcal{A}_x$ and $\mathcal{A}_y$ keep a state that may depend on the sequence of losses and decisions). The game is as follows:

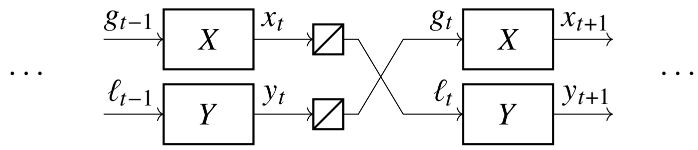

Figure 6.1 The flow of strategies and losses in regret minimization for games.

- Initialize $x_1 \in X, y_1 \in Y$ to be some pair of strategies in the relative interior (in matrix games we usually start with the uniform strategy)
- At time t, let x_t be the recommendation from $\mathcal{A}_x$ and y_t be the recommendation from $\mathcal{A}_y$
- Let $\mathcal{A}_x$ and $\mathcal{A}_y$ observe losses $g_t = f(\cdot, y_t), \ell_t = f(x_t, \cdot)$ respectively

For a strategy pair $\bar{x}, \bar{y}$, we will measure proximity to Nash equilibrium via the *saddle-point residual* (SPR):

$$\begin{aligned}\xi(\bar{x}, \bar{y}) &:= \left[\max_{y \in Y} f(\bar{x}, y) - f(\bar{x}, \bar{y})\right] + \left[f(\bar{x}, \bar{y}) - \min_{x \in X} f(x, \bar{y})\right] \\ &= \max_{y \in Y} f(\bar{x}, y) - \min_{x \in X} f(x, \bar{y}).\end{aligned}$$

Each bracketed term represents how much each player can improve by deviating from $\bar{y}$ or $\bar{x}$ respectively, given the strategy profile $(\bar{x}, \bar{y})$. In game-theoretic terms, the brackets capture how much each player improves by best responding.

Suppose that $\mathcal{A}_x$ and $\mathcal{A}_y$ guarantee regret bounds of the form

$$\begin{aligned}&\sum_{t \in [T]} f(x_t, y_t) - \min_{x \in X} \sum_{t \in [T]} f(x, y_t) \leq \epsilon_x, \\ &\max_{y \in Y} \sum_{t \in [T]} f(x_t, y) - \sum_{t \in [T]} f(x_t, y_t) \leq \epsilon_y.\end{aligned} \tag{6.1}$$

Then the following theorem holds.

Theorem 6.1 *Suppose* (6.1) *holds, then for the average strategies* $\bar{x} = \frac{1}{T}\sum_{t \in [T]} x_t, \bar{y} = \frac{1}{T}\sum_{t \in [T]} y_t$ *the SPR is bounded by*

$$\xi(\bar{x}, \bar{y}) \leq \frac{(\epsilon_x + \epsilon_y)}{T}.$$

Proof Summing the two inequalities in (6.1) we get

$$\begin{aligned}
\epsilon_x + \epsilon_y \geq & \max_{y\in Y} \sum_{t\in[T]} f(x_t, y) - \sum_{t\in[T]} f(x_t, y_t) + \sum_{t\in[T]} f(x_t, y_t) - \min_{x\in X} \sum_{t\in[T]} f(x, y_t) \\
= & \max_{y\in Y} \sum_{t\in[T]} f(x_t, y) - \min_{x\in X} \sum_{t\in[T]} f(x, y_t) \\
= & T \max_{y\in Y} \sum_{t\in[T]} \frac{1}{T} f(x_t, y) - T \min_{x\in X} \sum_{t\in[T]} \frac{1}{T} f(x, y_t) \\
\geq & T \left[\max_{y\in Y} f(\bar{x}, y) - \min_{x\in X} f(x, \bar{y}) \right],
\end{aligned}$$

where the second inequality is by f being convex-concave.

□

So now we know how to compute a Nash equilibrium: simply run the above repeated game with each player using a regret-minimizing algorithm, and the uniform average of the strategies will converge to a Nash equilibrium.

Figure 6.2 shows the performance of the various regret-minimization algorithms covered so far in the book, when used to compute a Nash equilibrium of a zero-sum matrix game via Theorem 6.1. Performance is shown on 3 randomized matrix game classes where entries in A are sampled according to: 100-by-100 uniform $[0, 1]$, 500-by-100 standard Gaussian, and 100-by-100 standard Gaussian. All plots are averaged across 50 game samples per setup (we do not show error bars because they are so small that they are hidden by the markers). We show one additional algorithm for reference: the *optimistic* variant of OMD, which is an "accelerated" variant of OMD that converges to a Nash equilibrium at a rate of $O\left(\frac{1}{T}\right)$. We cover OOMD and optimism in Chapter 7. The plot shows OOMD with the Euclidean distance.

As we see in Figure 6.2, OOMD indeed performs better than all the regret minimizers with a $O\left(\frac{1}{\sqrt{T}}\right)$ convergence-rate guarantee using the setup for Theorem 6.1. On the other hand, the entropy-based variant of OMD, which has a $\log n$ dependence on the dimension n, performs much worse than the algorithms with $\sqrt{n}$ dependence, even though the number of actions is on the order of hundreds.

6.2 Alternation

Next we introduce a minor change to the self-play setup called *alternation*. In alternation, the players are no longer symmetric: one player sees the loss based

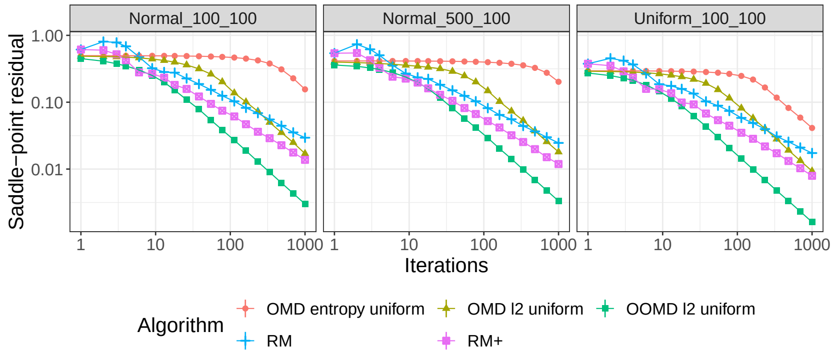

Figure 6.2 Plots showing the performance of regret matching, regret matching$^+$, OMD with the Euclidean DGF, and OMD with the entropy DGF for computing Nash equilibrium, all using Theorem 6.1. The optimistic variant of OMD (OOMD; see Chapter 7) with the Euclidean DGF is also presented.

on the previous strategy of the other player as before, but the second player sees the loss associated to the current strategy.

- Initialize x_1, y_1 to be uniform distributions over actions.
- At time t, let x_t be the recommendation from $\mathcal{A}_x$.
- The y player observes loss $f(x_t, \cdot)$.
- y_t is the recommendation from $\mathcal{A}_y$ after observing $f(x_t, \cdot)$.
- The x player observes loss $f(\cdot, y_t)$.

Suppose that the regret-minimizing algorithms guarantee regret bounds of the form

$$\begin{aligned} \max_{y \in Y} \sum_{t \in [T]} f(x_{t+1}, y) - \sum_{t \in [T]} f(x_{t+1}, y_t) &\le \epsilon_y, \\ \sum_{t \in [T]} f(x_t, y_t) - \min_{x \in X} \sum_{t \in [T]} f(x, y_t) &\le \epsilon_x. \end{aligned} \tag{6.2}$$

Theorem 6.2 *Suppose we run two regret minimizers with alternation, and they give the guarantees in (6.2). Then the average strategies* $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x_{t+1}, \bar{y} = \frac{1}{T}\sum_{t \in [T]} y_t$ *satisfy*

$$\xi(\bar{x}, \bar{y}) \le \frac{\epsilon_x + \epsilon_y + \sum_{t=1}^{T} \left(f(x_{t+1}, y_t) - f(x_t, y_t)\right)}{T}.$$

Proof As before we sum the regret bounds to get

$$\begin{aligned}\epsilon_x + \epsilon_y \geq &\max_{y\in Y}\sum_{t=1}^{T} f(x_{t+1}, y) - \sum_{t=1}^{T} f(x_{t+1}, y_t) + \sum_{t=1}^{T} f(x_t, y_t) - \min_{x\in X}\sum_{t=1}^{T} f(x, y_t)\\ =&\max_{y\in Y}\sum_{t=1}^{T} f(x_{t+1}, y) - \min_{x\in X}\sum_{t=1}^{T} f(x, y_t) - \sum_{t=1}^{T}[f(x_{t+1}, y_t) - f(x_t, y_t)]\\ \geq& T\left[\max_{y\in Y} f(\bar{x}, y) - \min_{x\in X} f(x, \bar{y})\right] - \sum_{t=1}^{T}[f(x_{t+1}, y_t) - f(x_t, y_t)].\end{aligned}$$

□

Theorem 6.2 shows that if $f(x_{t+1}, y_t) - f(x_t, y_t) \leq 0$ for all t, then the bound for alternation is weakly better than the bound in Theorem 6.1. But what does this condition mean? If we examine it from the regret minimization perspective, it is saying that x_{t+1} does better than x_t against y_t. Intuitively, we would expect this to hold: x_t is chosen right before observing $f(\cdot, y_t)$, whereas x_{t+1} is chosen immediately after observing $f(\cdot, y_t)$, and generally we would expect that any time we make a new observation, we should move somewhat in the direction of improvement against that observation. Indeed, it turns out to be relatively straightforward to show that this holds for all the regret minimizers we saw so far (As an exercise, show that this holds for a few regret minimizers; it is easiest for OMD).

Figure 6.3 shows the performance of the same set of regret-minimization algorithms but now using the setup from Theorem 6.2.

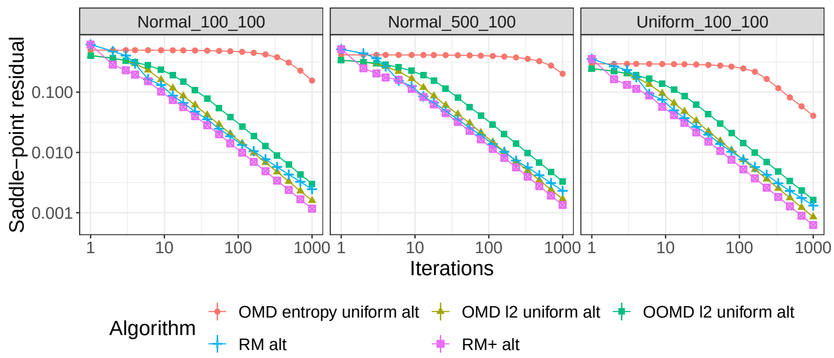

Figure 6.3 Plots showing the performance of four different regret-minimization algorithms for computing Nash equilibrium, all using Theorem 6.2. The optimistic variant of OMD (OOMD; see Chapter 7) is also presented.

Amazingly, Figure 6.3 shows that simply by adding alternation, OMD with the Euclidean DGF, regret matching, and RM$^+$ all perform about on par with OOMD, whereas they were noticeably worse before.

6.3 Increasing Iterate Averaging

Now we will look at one final trick. In Theorems 6.1 and 6.2 we generated a solution by uniformly averaging iterates. We will now consider polynomial averaging schemes of the form

$$\bar{x} = \frac{1}{\sum_{t\in[T]} t^q} \sum_{t\in[T]} t^q x_t, \quad \bar{y} = \frac{1}{\sum_{t\in[T]} t^q} \sum_{t\in[T]} t^q y_t.$$

Figure 6.4 shows the performance of the same set of regret-minimization algorithms but now using the setup from Theorem 6.2 and linear averaging in all algorithms. The fastest algorithm with uniform averaging, RM$^+$ with

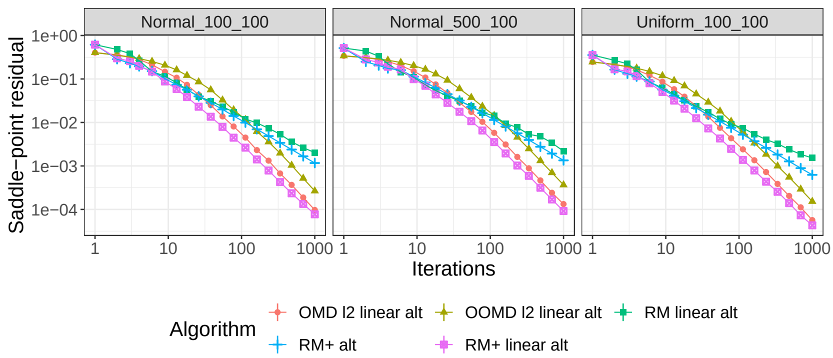

Figure 6.4 Plots showing the performance of four different regret-minimization algorithms for computing Nash equilibrium, all using Theorem 6.2. All algorithms use linear averaging. RM$^+$ with uniform averaging is shown as a reference point.

alternation, is shown for reference. OMD with Euclidean DGF and RM$^+$ with alternation both gain another order of magnitude in performance by introducing linear averaging.

It can be shown that RM$^+$, online mirror descent, and OOMD all continue to work with polynomial averaging schemes, in the sense that they have the same asymptotic rate of convergence as with uniform averaging. Interestingly, this is not the case for regret matching and FTRL, which do not work with the more aggressive averaging schemes.

6.4 Historical Notes

The derivation of a folk theorem for alternation in matrix games was by Burch et al. (2019), after Farina et al. (2019b) pointed out that the original folk theorem does not apply when using alternation. The general convex-concave case is new, although easily derived from the existing results.

The fact that instantiating OMD with the Euclidean distance seems to perform better than entropy when solving matrix games in practice has been observed in a few different algorithms both first-order methods (Chambolle and Pock, 2016; Gao et al., 2021a) and regret-minimization algorithms (Farina et al., 2019c). The fact that OMD with Euclidean distance performs much better after adding alternation has not been observed before.

Results for polynomial averaging schemes were shown by Tammelin et al. (2015) and Brown and Sandholm (2019a) for RM^+, in Nemirovski's lecture notes[1] for mirror descent and mirror prox, and for several other primal-dual first-order methods by Gao et al. (2021a).

Further reading.

Self play in games via regret minimization as a computational tool is not covered in other books as far as I know. From a more theoretical perspective, Cesa-Bianchi and Lugosi (2006) covers some self play results. The best sources for further reading would be papers such as Tammelin et al. (2015); Farina et al. (2021b). The PhD thesis of Neil Burch (Burch, 2018) also has a lot of interesting results and numerics.

[1] https://www2.isye.gatech.edu/~nemirovs/LMCO_LN2019NoSolutions.pdf

7
Optimism and Fast Convergence of Self Play

In this chapter we study *optimistic* variants of online learning. The idea is that we have some prediction about the next loss at each time step, and we want to see how well we can do when using predictions (while ensuring good performance when the predictions are wrong). This will yield the optimistic online mirror descent algorithm that we showed in some plots in Chapter 6. It will also yield an example of an online learning approach that converges to a Nash equilibrium of a two-player zero-sum game at a rate of $O(1/T)$, as opposed to the $O(1/\sqrt{T})$ rate we saw previously.

7.1 Predictive Online Learning

Suppose that we are in an online learning setting as in Chapter 4: we must repeatedly choose actions $x_t \in X \subset \mathbb{R}^n$ for some convex and compact decision set X, and then we receive (linear) losses $g_t \in \mathbb{R}^n$. But now suppose we receive some additional information about the loss function g_t before we have to make a prediction. In particular, we will suppose that at each time t, we are given some *prediction* $m_t \in [0,1]^n$ of the loss g_t. Formally, we now have the following learning protocol: at each time step $t = 1, \ldots, T$:

(i) We are given a prediction $m_t \in [0,1]^n$.
(ii) We must choose a decision $x_t \in X$.
(iii) Afterwards, a loss vector $g_t \in [0,1]^n$ is revealed to us, and we pay the loss $\langle g_t, x_t \rangle$.

The question is to what extent we can use the prediction to do better than in the standard online learning setting. It is immediately clear that we have to be careful about what we want. Suppose that the predictions are perfect, i.e. $m_t = g_t$, then we can simply best respond to m_t, i.e. select $x_t = \arg\min_{x \in \Delta^n} \langle m_t, x \rangle$, and we will do as well as the best sequence of decisions in hindsight, and generally have significant *negative* regret against the single best action in hindsight. On the other hand, if m_t turns out to be inaccurate, then best responding to m_t could yield linear regret. Ideally, we would like regret guarantees that degrade gracefully with the accuracy of m_t, while still doing "well" when m_t is a good prediction. Next we will show that this is indeed possible, with variations on the OMD and FTRL algorithms introduced in Chapter 4.

7.1.1 Online Mirror Descent with Predictions

First we consider OMD with predictions. OMD with predictions is usually called *optimistic online mirror descent* (OOMD). There are two ways to incorporate the prediction m_t into the OMD algorithm. The first is what we will call single-step OOMD:

$$x_{t+1} = \arg\min_{x \in X} \langle g_t + m_{t+1} - m_t, x \rangle + \frac{1}{\eta} D(x \| x_t).$$

As a base case, let $x_0 = \arg\min_{x \in X} d(x)$ and $m_0 = 0$. Intuitively, we can think of $g_t - m_t$ as "undoing" the previous move in the direction of m_t and instead moving in the direction of g_t. Then, we additionally "optimistically" move in the direction of m_{t+1}.

We now show that single-step OOMD satisfies a regret bound that lets us get compelling guarantees whether the predictions are accurate or not.

Theorem 7.1 *Assume that $m_1 = 0$ and d is 1-strongly convex. The regret of single-step OOMD with respect to a sequence of losses $g_1, \dots, g_T$ and predictions $m_1, \dots, m_T$ is bounded by*

$$R_T \le \frac{D(x \| x_1)}{\eta} + \eta \sum_{t \in [T]} \|g_t - m_t\|_*^2 - \frac{1}{4\eta} \sum_{t \in [T]} \|x_{t+1} - x_t\|^2.$$

Proof By first-order optimality, we have for each $t \in \{1, \dots, T\}$ that

$$\begin{aligned} &\langle m_{t+1} + g_t - m_t + (1/\eta)\nabla d(x_{t+1}) - (1/\eta)\nabla d(x_t), x - x_{t+1} \rangle \ge 0 \\ \Leftrightarrow\ &\langle m_{t+1} + g_t - m_t, x_{t+1} - x \rangle \le \frac{1}{\eta} \langle \nabla d(x_{t+1}) - \nabla d(x_t), x - x_{t+1} \rangle. \end{aligned}$$

Applying the three-point lemma (Lemma 4.4) we get

$$\langle m_{t+1} + g_t - m_t, x_{t+1} - x \rangle \le \frac{1}{\eta} \left(D(x \| x_t) - D(x \| x_{t+1}) - D(x_{t+1} \| x_t) \right). \quad (7.1)$$

Summing Eq. (7.1) over $t = 1, \ldots, T$ and removing telescoping terms on both sides, we get

$$\sum_{t\in[T]} \langle g_t, x_{t+1} - x\rangle + \sum_{t\in[T]} \langle m_{t+1} - m_t, x_{t+1}\rangle + \langle m_1 - m_{T+1}, x\rangle \le \frac{1}{\eta}\left(D(x\|x_1) - D(x\|x_{T+1}) - \sum_{t\in[T]} D(x_{t+1}\|x_t)\right). \tag{7.2}$$

Now we simplify the left-hand side of Eq. (7.2).

$$\begin{aligned}
&\sum_{t\in[T]} \langle g_t, x_{t+1} - x\rangle + \sum_{t\in[T]} \langle m_{t+1} - m_t, x_{t+1}\rangle + \langle m_1 - m_{T+1}, x\rangle \\
&= \sum_{t\in[T]} \langle g_t, x_{t+1} - x\rangle + \sum_{t\in[T]} \langle m_{t+1} - m_t, x_{t+1}\rangle \\
&= \sum_{t\in[T]} \langle g_t, x_t - x\rangle + \sum_{t\in[T]} \langle g_t - m_t, x_{t+1} - x_t\rangle + \sum_{t\in[T]} \langle m_{t+1}, x_{t+1}\rangle - \sum_{t\in[T]} \langle m_t, x_t\rangle \\
&= \sum_{t\in[T]} \langle g_t, x_t - x\rangle + \sum_{t\in[T]} \langle g_t - m_t, x_{t+1} - x_t\rangle + \langle m_{T+1}, x_{T+1}\rangle - \langle m_1, x_1\rangle \\
&= \sum_{t\in[T]} \langle g_t, x_t - x\rangle + \sum_{t\in[T]} \langle g_t - m_t, x_{t+1} - x_t\rangle.
\end{aligned} \tag{7.3}$$

The first step is by noting that we set $m_1 = 0$, and we can assume $m_{T+1} = 0$ without changing the regret up to time T. The second step is by adding and subtracting $\langle g_t, x_t\rangle + \langle m_t, x_t\rangle$ for each t. The third step is by telescoping terms. The fourth step is again by noting that we set $m_1 = 0$ and $m_{T+1} = 0$.

Combining Eq. (7.2) and Eq. (7.3), we get

$$\begin{aligned}
\sum_{t\in[T]} \langle g_t, x_t - x\rangle \le &\sum_{t\in[T]} \langle g_t - m_t, x_t - x_{t+1}\rangle \\
&+ \frac{1}{\eta}\left(D(x\|x_1) - D(x\|x_{T+1}) - \sum_{t\in[T]} D(x_{t+1}\|x_t)\right).
\end{aligned} \tag{7.4}$$

Notice that the left-hand side is the regret up to time T. Next we simplify the first term on the right-hand side via the Cauchy-Schwarz inequality and the Peter-Paul inequality (see Eq. (A.3)):

$$\begin{aligned}
\langle g_t - m_t, x_t - x_{t+1}\rangle &\le \|g_t - m_t\|_* \|x_t - x_{t+1}\| \\
&\le \eta\|g_t - m_t\|_*^2 + \frac{1}{4\eta}\|x_t - x_{t+1}\|^2.
\end{aligned}$$

Plugging this upper bound into Eq. (7.4) and using $D(x_{t+1}\|x_t) \ge \frac{1}{2}\|x_{t+1} - x_t\|^2$

(see Eq. (A.8) in Appendix A.4) we get the desired result.

$$\begin{aligned}\sum_{t\in[T]} \langle g_t, x_t - x\rangle \leq &\sum_{t\in[T]} \left(\eta\|g_t - m_t\|_*^2 + \frac{1}{4\eta}\|x_t - x_{t+1}\|^2 - \frac{1}{2\eta}\|x_{t+1} - x_t\|^2\right)\\ &+ \frac{1}{\eta}\left(D(x\|x_1) - D(x\|x_{T+1})\right)\\ \leq &\frac{1}{\eta}D(x\|x_1) + \sum_{t\in[T]}\left(\eta\|g_t - m_t\|_*^2 - \frac{1}{4\eta}\|x_{t+1} - x_t\|^2\right).\end{aligned}$$

□

The second way to incorporate predictions in OMD is the *two-step* OOMD algorithm. In two-step OOMD, we maintain two separate sequences of decisions:

$$x_{t+1} = \arg\min_{x\in X}\langle m_{t+1}, x\rangle + \frac{1}{\eta}D(x\|z_t),$$

$$z_{t+1} = \arg\min_{x\in X}\langle g_t, x\rangle + \frac{1}{\eta}D(x\|z_t).$$

Intuitively, we can think of z_t as the sequence of iterates generated by always moving in the direction of improvement against the losses $g_1, \dots g_t$, while each x_t is generated by taking one step in the direction of m_t from the previous iterate z_{t-1}. Because the steps in the direction of m_t are never incorporated into the sequence z_t, there is no need to "undo" moves as in single-step OOMD. Two-step OOMD is arguably less attractive than single-step OOMD, because it requires an additional proximal step. Two-step OOMD has the same regret guarantee as single-step OOMD.

The two-step OOMD procedure was the first to be introduced in the literature, and it was historically referred to simply as OOMD. In the rest of the book, when we refer to OOMD, it can be thought of as either the single-step or two-step procedure. For theoretical purposes, there is usually no difference. In practice single-step OOMD may be preferable, since it avoids the need for an additional proximal step.

7.2 Optimism and RVU Bounds

Next we study a particular form of prediction: we will use the *previous* loss as the prediction of the next loss. In particular, this means that we set $m_t = g_{t-1}$. Now, we are effectively saying that our predictions will be good if losses are not changing too rapidly over time. This leads to the notion of *Regret bounded by Variation in Utilities* (RVU):

Definition 7.2 An online learning algorithm satisfies the *Regret bounded by Variation in Utilities* (RVU) property with parameters $\alpha > 0, 0 < \beta \leq \gamma$ and a pair of primal-dual norms $\|\cdot\|, \|\cdot\|_*$ if its regret on a sequence of losses $g_1, \ldots, g_T$ is bounded by

$$R_T \leq \alpha + \beta \sum_{t \in [T]} \|g_t - g_{t-1}\|_*^2 - \gamma \sum_{t \in [T]} \|x_t - x_{t-1}\|^2.$$

If we instantiate OOMD with $m_t = g_{t-1}$, then Theorem 7.1 shows that OOMD satisfies the RVU property with parameters $\alpha = \max_{x \in X}(D(x\|x_1)/\eta$, $\beta = \eta$, and $\gamma = 1/(4\eta)$. Note that the sum over $\|x_{t+1} - x_t\|^2$ in Theorem 7.1 (known as the *path length*) does not include $\|x_1 - x_0\|^2$, but this value is zero, since $m_1 = g_0 = 0$.

7.3 Fast Convergence in Zero-Sum Games

Now we show that the RVU bounds can be used to obtain fast convergence in two-player zero-sum games. In particular, suppose that we have a game $\min_{x \in X} \max_{y \in Y} x^\top A y$ where X, Y are convex and compact, and A has operator norm $\|A\| \leq L$ with respect to the norms $\|\cdot\|_x, \|\cdot\|_y$. Suppose also that we have distance-generating functions d_x, d_y that are each 1-strongly convex with respect to $\|\cdot\|_x$, and $\|\cdot\|_y$.

Before we start studying the repeated game setup, it will be useful to derive a few inequalities that will allow us to relate A to the variation in the dual norm of losses $\|A(y_t - y_{t-1})\|_{x,*}$ and $\|-A^\top(x_t - x_{t-1})\|_{y,*}$. By definition of the operator norm, we have

$$\|A\| = \max_{\|x\|_x=1} \|A^\top x\|_{y,*} = \max_{\|x\|_x=1} \max_{\|y\|_y=1} x^\top A y,$$

$$\|A\| = \max_{\|x\|_x=1} \|A^\top x\|_{y,*} = \max_{x} \frac{1}{\|x\|_x} \|A^\top x\|_{y,*} \geq \frac{1}{\|x'\|_x} \|A^\top x'\|_{y,*} \ \forall x' \in X, \tag{7.5}$$

$$\|A\| = \max_{\|y\|_y=1} \|Ay\|_{x,*} = \max_{y} \frac{1}{\|y\|_y} \|Ay\|_{x,*} \geq \frac{1}{\|y'\|_y} \|Ay'\|_{x,*} \ \forall y \in Y. \tag{7.6}$$

The repeated game is as follows:

- Initialize $x_0 \in X, y_0 \in Y$ to be some pair of strategies in the relative interior (in matrix games we usually start with the uniform strategy).
- Provide a recommendation $m_t^x = Ay_{t-1}$ to $\mathcal{A}_x$ and $m_t^y = -A^\top x_{t-1}$ to $\mathcal{A}_y$.

- At time t, let x_t be the recommendation from $\mathcal{A}_x$ and y_t be the recommendation from $\mathcal{A}_y$.
- Let $\mathcal{A}_x$ and $\mathcal{A}_y$ observe losses $g_t = Ay_t, \ell_t = -A^\top x_t$ respectively.

In this setup, OOMD satisfies the RVU property with parameters $\alpha = (\max_{x\in X} D(x\|x_1)/\eta$, $\beta = \eta$, and $\gamma = 1/(4\eta)$, as described in the previous section.

Theorem 7.3 *Suppose that $x_1, \dots, x_T$ and $y_1, \dots, y_T$ are generated by regret minimizers satisfying the RVU property with parameters $\alpha_x, \beta_x, \gamma_x, \alpha_y, \beta_y, \gamma_y$ such that $\beta_x\|A\|^2 \le \gamma_y$ and $\beta_y\|A\|^2 \le \gamma_x$, then we have the following convergence rate for the pair of average strategies $\bar{x} = \frac{1}{T}\sum_{t\in[T]} x_t$ and $\bar{y} = \frac{1}{T}\sum_{t\in[T]} y_t$:*

$$\xi(\bar{x}, \bar{y}) \le \frac{\alpha_x + \alpha_y}{T}.$$

Proof We have

$$\begin{aligned}
T\xi(\bar{x}, \bar{y}) &= T\left(\max_y \langle Ay, \bar{x}\rangle - \min_x \langle A\bar{y}, x\rangle\right) \\
&= \max_y \sum_{t\in[T]} \langle Ay, x_t\rangle - \min_x \sum_{t\in[T]} \langle Ay_t, x\rangle \\
&= \max_y \sum_{t\in[T]} \langle Ay, x_t\rangle - \sum_{t\in[T]} \langle Ay_t, x_t\rangle + \sum_{t\in[T]} \langle Ay_t, x_t\rangle - \min_x \sum_{t\in[T]} \langle Ay_t, x\rangle \\
&\le \alpha_y + \beta_y \sum_{t\in[T]} \|A^\top(x_t - x_{t-1})\|_*^2 - \gamma_y \sum_{t\in[T]} \|y_t - y_{t-1}\|^2 \\
&\quad \alpha_x + \beta_x \sum_{t\in[T]} \|A(y_t - y_{t-1})\|_*^2 - \gamma_x \sum_{t\in[T]} \|x_t - x_{t-1}\|^2. \qquad (7.7)
\end{aligned}$$

The second equality is by expanding $\bar{x}$ and $\bar{y}$. The inequality follows by noting that we have the sum of the player regrets, and then applying the RVU bound. Now we upper bound Eq. (7.7) by using Eqs. (7.5) and (7.6) to get

$$\begin{aligned}
Eq.\ (7.7) &\le \alpha_y + \beta_y\|A\|^2 \sum_{t\in[T]} \|x_t - x_{t-1}\|^2 - \gamma_y \sum_{t\in[T]} \|y_t - y_{t-1}\|^2 \\
&\quad \alpha_x + \beta_x\|A\|^2 \sum_{t\in[T]} \|y_t - y_{t-1}\|^2 - \gamma_x \sum_{t\in[T]} \|x_t - x_{t-1}\|^2 \\
&= \alpha_y + \alpha_x + (\beta_y\|A\|^2 - \gamma_x) \sum_{t\in[T]} \|x_t - x_{t-1}\|_*^2 \\
&\quad + (\beta_x\|A\|^2 - \gamma_y) \sum_{t\in[T]} \|y_t - y_{t-1}\|_*^2
\end{aligned}$$

Finally, using $\beta_x \|A\|^2 \leq \gamma_y$ and $\beta_y \|A\|^2 \leq \gamma_x$ we get $T\xi(\bar{x}, \bar{y}) \leq \alpha_y + \alpha_x$. Dividing everything by T yields the result. □

Corollary 7.4 *Suppose that $x_1, \ldots, x_T$ and $y_1, \ldots, y_T$ are generated by OOMD with stepsizes $\eta_x \leq 1/(2\|A\|), \eta_y \leq 1/(2\|A\|)$ with the previous loss as the prediction, then we have the following convergence rate for the pair of average strategies $\bar{x} = \frac{1}{T}\sum_{t\in[T]} x_t$ and $\bar{y} = \frac{1}{T}\sum_{t\in[T]} y_t$:*

$$\xi(\bar{x}, \bar{y}) \leq \frac{\max_{x\in X} D(x\|x_1)}{\eta_x T} + \frac{\max_{y\in Y} D(y\|y_1)}{\eta_y T}.$$

7.4 Small Individual Regrets in General-Sum Games

Next we show that the RVU bounds can be used to obtain small individual regrets in general-sum games. This will rely on each algorithm having a *stability* property, meaning that the algorithm's recommendation does not change too much between each time step.

Lemma 7.5 *The decisions of OOMD are* stable *in the sense that $\|x_{t+1} - x_t\| \leq \eta\|m_{t+1} + g_t - m_t\|_*$. Suppose $m_t = g_{t-1}$, then we have $\|x_{t+1} - x_t\| \leq \eta\|2g_t - g_{t-1}\|_*$.*

Proof Since $D(\cdot\|x)$ is 1-strongly convex for any x, we have that its convex conjugate is 1-Lipschitz with respect to its gradient. The iterates x_t and x_{t+1} are respectively equal to the gradients of the convex conjugate $D^*(\cdot\|x_t)$ at 0 and at $\eta(g_t + m_{t+1} - m_t)$. Thus, we have $\|x_{t+1} - x_t\| \leq \eta\|g_t + m_{t+1} - m_t\|_*$. □

Consider a general-sum game where we have n players, decision spaces X_i, and each player has a concave utility function $u_i(x)$ which is Lipschitz in the sense that $\|\nabla u_i(x) - \nabla u_i(x')\|_* \leq L_i \sum_{j=1}^n \|x_j - x'_j\|$ for some $L_i > 0$. This is satisfied e.g. if u_i is multilinear, as in the case of normal-form games and extensive-form games. The repeated game is as follows:

- Initialize $x_0^i \in X_i$ to be a strategy in the relative interior for each player i (in normal-form games we usually start with the uniform strategy)
- Provide a recommendation $m_t^i = \nabla u_i(x_{t-1})$ to the regret minimizer for each player i
- At time t, let x_t^i be the recommendation for player i and x_t be the collection of recommendations for all players (i.e. the strategy profile at time t).
- Let player i observe the loss $g_t^i = \nabla u_i(x_t)$

As in the previous section, OOMD satisfies the RVU property with parameters $\alpha = (D(x\|x_1)/\eta$, $\beta = \eta$, and $\gamma = 1/(4\eta)$.

Theorem 7.6 *Suppose that each player's decisions i in a general-sum game are* stable *in the sense that* $\|x^i_t - x^i_{t-1}\| \le \kappa$ *for all t, and each player uses a regret minimizer with RVU guarantees* $\alpha_i, \beta_i, \gamma_i$*. Then each player's regret is bounded as follows*

$$R^i_T \le \alpha_i + \beta_i T L_i^2 n^2 \kappa^2.$$

Proof First note that from Lipschitzness of the game, we have

$$\begin{aligned}\sum_{t\in[T]} \|g^i_t - g^i_{t-1}\|^2_* &\le \sum_{t\in[T]} L_i^2 \left(\sum_{j=1}^n \|x^j_t - x^j_{t-1}\| \right)^2 \\ &\le \sum_{t\in[T]} L_i^2 \left(\sum_{j=1}^n \kappa \right)^2 \\ &\le T L_i^2 n^2 \kappa^2.\end{aligned}$$

Combining this with the RVU property, we have

$$\begin{aligned}R^i_T &\le \alpha_i + \beta_i \sum_{t\in[T]} \|g^i_t - g^i_{t-1}\|^2_* - \gamma_i \sum_{t\in[T]} \|x^i_t - x^i_{t-1}\|^2 \\ &\le \alpha_i + \beta_i T L_i^2 n^2 \kappa^2.\end{aligned}$$

□

Now we immediately get a better than $\sqrt{T}$ regret bound for OOMD by setting the stepsize the right way.

Corollary 7.7 *Suppose that each player's decisions are generated by OOMD with stepsizes* $\eta_i = \Omega_i^{1/4}/(T^{1/4} L_i^{1/2} n^{1/2})$*, then each player's regret is bounded as follows*

$$R^i_T \le 2\Omega_i^{3/4} T^{1/4} L_i^{1/2} n^{1/2}.$$

Proof Instantiating the regret bound with OOMD gives

$$R^i_T \le \frac{\Omega_i}{\eta} + \eta^3 T L_i^2 n^2 \le \Omega_i^{3/4} T^{1/4} L_i^{1/2} n^{1/2} + \Omega_i^{3/4} T^{1/4} L_i^{1/2} n^{1/2}.$$

□

7.5 Historical Notes

The idea of predictive online learning leading to fast convergence in zero-sum games was shown by Rakhlin and Sridharan (2013). The formulation of RVU bounds was given by Syrgkanis et al. (2015), where they showed that the bounds can be used to obtain fast convergence in two-player zero-sum games, and improved regret bounds in general-sum games. Earlier, Daskalakis et al. (2015) (while the final journal paper was published in 2015, the conference version of that work appeared in 2011) had showed that it is possible to achieve $O(\ln T/T)$ convergence in two-player zero-sum games via self-play with no-regret learning dynamics, but their result relied on a somewhat intricate learning dynamic based on a decentralized implementation of the EGT algorithm for saddle-point problems (Nesterov, 2005a).

The idea of optimism and fast convergence in two-player zero-sum games is also related to earlier works in the first-order methods literature, where some form of *extrapolation* leads to an $O(1/T)$ rate of convergence for convex-concave saddle-point problems. For example, the mirror prox method by Nemirovski (2004) achieves this rate, and as pointed out by Rakhlin and Sridharan (2013), optimistic OMD in self play can be seen as achieving a similar idea as mirror prox. Moreover, in the case of using the Euclidean DGF in optimistic OMD for solving a two-player zero-sum game, the algorithm is equivalent to an algorithm given by Popov (1980), though the $O(1/T)$ rate was not known at the time. Prior to the $O(1/T)$ rate result by Nemirovski (2004), Nesterov was, to the best of my knowledge, the first to show that such rates are attainable via first-order methods. Nesterov's approach used what's now known as *Nesterov smoothing* (Nesterov, 2005b), where a smooth approximation to the nonsmooth problem is constructed, and then this approximation is solved via accelerated first-order methods. Though the Nesterov smoothing paper appeared in a journal in 2005 and the Nemirovski paper appeared in 2004, the Nesterov paper predates the Nemirovski paper; it was made available online in 2003. In fact, Nemirovski explicitly credits Nesterov's work as an inspiration in his paper. The inversion of dates is due to the tardiness of the journal publication process. Concurrently with Nemirovski's mirror prox result, Nesterov also developed the *excessive gap technique* (EGT), another method that achieves $O(1/T)$ via first-order updates (Nesterov, 2005a).

Optimism in EFGs was first studied by Farina et al. (2019c), where they use dilated distance-generating functions (DGFs) such as those we studied in Section 8.4. However, the numerical performance turned out to be worse than that of CFR^+ algorithms. Lee et al. (2021) showed last-iterate convergence results for optimistic algorithms in two-player zero-sum EFGs that use dilated DGFs, though with the assumption of a unique Nash equilibrium in the case of dilated entropy-based DGFs.

Based on the strong practical performance of CFR^+ compared to optimistic methods in EFG solving, it was a natural question whether "optimistic learning" in CFR^+ is possible. Farina et al. (2021b) and Flaspohler et al. (2021) concurrently showed how to design predictive variants of RM^+. Farina et al. (2021b) introduced predictive CFR^+ which combines CFR and predictive RM^+. They show that predictive CFR^+ leads to very strong practical performance in many games. Interestingly, they found that non-predictive CFR^+ is faster for poker games, whereas predictive CFR^+ is *much* faster for various non-poker EFG benchmark games. However, no theoretical improvement over non-predictive CFR^+ or RM^+ is achieved by these algorithms, in terms of dependence on the number of iterations T when used in self play in two-player zero-sum games. Unlike for OMD, OOMD, and various FTRL variants, it was recently shown that the RM+ algorithm is not stable (Farina et al., 2023). This is a key reason why the predictive variant of RM^+ does not achieve a $1/T$ convergence rate in zero-sum games (in theory), since it means that the previous loss is not always a good prediction of the next loss. Farina et al. (2023) also show numerical examples where predictive RM^+ converges at a rate of $1/\sqrt{T}$.

Further reading.

Optimism is too recent to have extensive textbook coverage. Orabona (2019) has some good coverage of optimism. In a game-solving context, I recommend reading Syrgkanis et al. (2015) for a well-written paper that introduced RVU bounds and shows a lot of useful results that can be developed from those RVU bounds. Farina et al. (2021b) is a good paper to read for the use of optimism in EFG solving.

8
Extensive-Form Games

In this chapter we will cover *extensive-form games* (EFGs). Extensive-form games are a richer game description than normal-form games that explicitly models sequential interaction and chance (such as the dealing of cards). EFGs are played on a game tree. Each node in the game tree belongs to some player, and that player gets to choose the branch to traverse. Superhuman poker AIs were created in large part through the design of good algorithms for computing (approximate) Nash equilibria in EFGs.

8.1 Perfect-Information EFGs

We start by considering *perfect-information* EFGs. The term perfect information refers to the fact that in these games, every player always knows the exact state of the game. A perfect-information EFG is a game played on a tree, where each internal node belongs to some player. The actions for the player at a given node is the set of branches, and by selecting a branch the game proceeds to the following node. An example is shown in Figure 8.1 on the left. That game has four nodes where players take actions, two belong to player 1 (labelled P1) and two belonging to player 2 (labelled P2). Additionally, the game tree has 6 leaf nodes. At each leaf node, each player receives some payoff. In this particular game, it is a zero-sum game, and the value at a leaf denotes the value that player 1 receives.

Perfect-information EFGs are trivially solvable (at least if we are able to traverse the whole game tree at least once). The way to solve them is via *backward induction*. Backward induction works by starting at some bottom decision node of the game tree, which only leads to leaf nodes after each action is taken (such a node always exists). Then, the optimal action for the player at the node is selected (which can be done by choosing the one that maximizes their

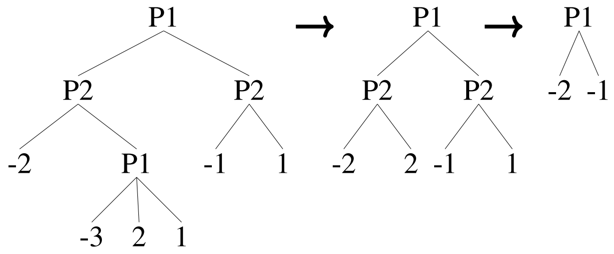

Figure 8.1 A simple perfect-information EFG. Three versions of the game are shown, where each stage corresponds to removing one layer of the game via backward induction.

utility), and the decision node is replaced with the corresponding leaf node. Now we have a new perfect-information EFG with one less decision node. Backward induction then repeats this process until there's no internal nodes left, at which point we have computed a Nash equilibrium, since every player acted optimally at every decision node throughout the backward induction. This immediately shows that perfect-information EFGs always have pure-strategy Nash equilibria.

While backward induction yields a linear-time algorithm for solving perfect-information games, in practice, many games of interest are way too large to solve with it nonetheless. For example, chess and go both have enormous game trees, with estimates of $\sim 10^{45}$ and $\sim 10^{172}$ nodes respectively. In such cases, tree search methods such as Monte Carlo Tree Search (MCTS) are used. Because our focus will be on the general class of EFGs without perfect information, we do not go into MCTS.

An EFG can always be represented as a normal-form game. Intuitively, an action in the corresponding normal-form game should specify what the player does at *every* decision point in the EFG. Thus, we create an action corresponding to every possible way of assigning an action at every decision point. So, if a player has d decision points with A actions each, then there are A^d actions in the normal form representation of the EFG. Clearly this is not efficient from a computational perspective, as the NFG representation is exponentially-large in the size of the EFG representation. Nonetheless, this reduction is a useful tool that can sometimes be used both algorithmically and theoretically. This reduction to normal form works for both perfect and imperfect-information games.

Let's consider an instructive example. We will model the Cuban Missile Crisis. The USSR has moved a bunch of nuclear weapons to Cuba, and the US has to decide how to respond. If they do nothing, then the USSR wins a political

		USSR	
		Nuclear war	Compromise
USA	Respond	−1000, −1000	2,1
	Do Nothing	0,2	0,2

Table 8.1 *The normal-form payoff matrix for the Cuban Missile Crisis game.*

victory, and gets to keep nuclear missiles within firing distance of major US cities. If the US responds, then it could result in a series of escalations that would eventually lead to nuclear war, or the USSR will eventually compromise and remove the missiles. Suppose the payoff are as follows:

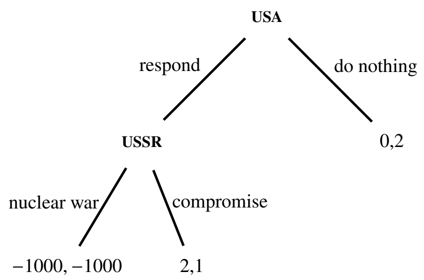

Figure 8.2 A perfect-information EFG modeling the Cuban missile crisis.

If we convert this game to normal form, we get the following game:

It is straightforward to see from this representation that the Cuban Missile Crisis game has two PNE: (do nothing, nuclear war) and (respond, compromise). However, the first PNE is in a sense not compelling: what if the USA just responded? The USSR probably would not be willing to follow through on taking the action "nuclear war" since it has such low utility for them as well. This leads to the notion of *subgame perfect equilibria*, which are equilibria that remain equilibria if we take any *subgame* consisting of picking some node in the tree and starting the game there.

8.2 Imperfect-Information EFGs

Next we study the more general class of EFGs which include imperfect information. These are games played on a tree again, but where players may not have perfect knowledge about the state of the game. From a game-theoretic perspective, this class of games is richer, and will rely more directly on equilibrium concepts for talking about solutions (in contrast to perfect-information EFGs, where solutions are straightforwardly obtained from backward induction). An example is shown in Figure 8.3.

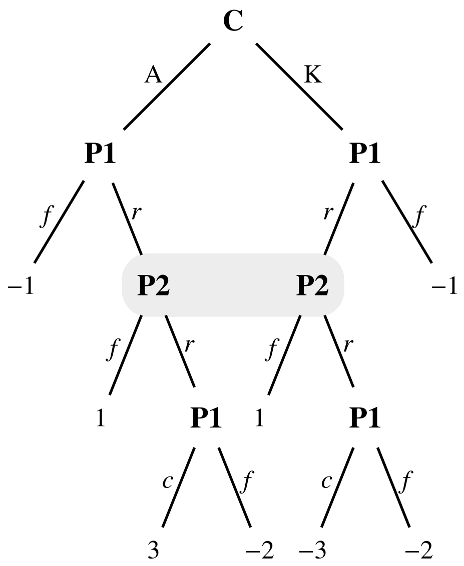

Figure 8.3 A poker game where P1 is dealt Ace or King with equal probability ('C' denotes a chance node). "r," "f," and "c" stands for raise, fold, and check respectively. Leaf values denote P1 payoffs. The shaded area denotes an information set: P2 does not know which of these nodes they are at, and must thus use the same strategy in both.

An EFG has the following:

- Information sets (sometimes shortened to "infosets"): for each player, the nodes belonging to that player are partitioned into *information sets* $I \in \mathcal{I}_i$. Information sets represent imperfect information: a player does not know which node in an information set they are at, and thus they must utilize the same strategy at each node in that information set. In Figure 8.3 P2 has only 1 information set, which contains both their nodes, whereas P1 has four information sets, each one a singleton node. For player i we will also let $\mathcal{J}_i$ be an index set of information sets with generic element j.
- Each information set I with index j has a set of actions that the corresponding player may take, which is denoted by A_j.

- Leaf nodes Z: the set of terminal states. Player i gains utility $u_i(z)$ if leaf node z is reached. Z is the set of all leaf nodes.
- Chance nodes where Chance or Nature moves with a fixed probability distribution. In Figure 8.3 chance deals A or K with equal probability.

We will assume throughout that the game has *perfect recall*, which means that no player ever forgets something they knew in the past. More formally, it means that for every information set $I \in \mathcal{I}_i$, there is a single last information-set action pair I', a' belonging to i that was the last information set and action taken by that player for every node in I.

The last action taken by player i before reaching an information set with index j is denoted p_j. This is well-defined due to perfect recall.

In Chapter 6 we saw how to compute a Nash equilibrium in a two-player zero-sum game by finding a saddle point of a min-max problem over convex compact polytopes. This model looked as follows:

$$\min_{x \in X} \max_{y \in Y} \langle x, Ay \rangle. \tag{8.1}$$

Now we would like to find a way to represent two-player zero-sum EFGs in this way. This turns out to be possible, and the key is to find the right way to represent strategies such that we get a bilinear objective. The next section will describe this representation.

First, let us see why the most natural formulation of the strategy spaces won't work. The natural formulation would be to have a player specify a probability distribution over actions at each of their information sets. Such strategies are called *behavioral strategies*. Let σ be a strategy profile, where σ_a is the probability of taking action a (from now on we assume that every action is distinct so that for any a there is only one corresponding I where the action can be played). The expected value over leaf nodes is

$$\sum_{z \in Z} u_2(z) \mathbb{P}(z|\sigma).$$

The problem with this formulation is that if a player has more than one action on the path to any leaf, then the probability $\mathbb{P}(z|\sigma)$ of reaching z is non-convex in that player's own strategy, since we have to multiply each of the probabilities belonging to that player on the path to z. Thus, we cannot get the bilinear form in (8.1).

8.3 Sequence Form

In this section we will describe how we can derive a bilinear representation X of the strategy space for player 1. Everything is analogous for Y.

In order to get a bilinear formulation of the expected value we do not write our strategy in terms of the probability σ_a of playing an action a. Instead, we associate to each information-set-action pair I, a a variable x_a denoting the probability of playing the *sequence* of actions belonging to player 1 on the path to I, including the probability of a at I. For example, in the poker game in Fig. 8.3, there would be a variable $x_{\hat{c}}$ denoting the product of probabilities player 1 puts on playing actions r and then $\hat{c}$. To be concrete, say that we have a behavioral strategy σ^1 for player 1 in the game of Fig. 8.3, then the corresponding sequence-form probability on the action $\hat{c}$ would be $x_{\hat{c}} = \sigma^1_r \cdot \sigma^1_{\hat{c}}$. Similarly, there would be a variable $x_{\hat{f}} = \sigma^1_r \cdot \sigma^1_{\hat{f}}$ denoting the product of probabilities on r and $\hat{f}$. Clearly, for this to define a valid strategy we must have $x_{\hat{c}} + x_{\hat{f}} = x_r$.

More generally, X is defined as the set of all $x \in \mathbb{R}^n, x \geq 0$ such that

$$x_{p_j} = \sum_{a \in A_j} x_a, \forall j \in \mathcal{J}_1, \tag{8.2}$$

where $n = \sum_{I \in \mathcal{I}_i} |A|$, and $p(I)$ is the parent sequence leading to I.

One way to visually think of the set of sequence-form strategies is given in Figure 8.4. This representation is called a *treeplex*. Each information set is represented as a simplex, which is scaled by the parent sequence leading to that information set (by perfect recall there is a unique parent sequence). After taking a particular action it is possible that a player may arrive at several next possible simplexes depending on what the other players or nature does. This corresponds to the player observing information (e.g. which cards were dealt in a poker game, or whether the other player bets or checks). This is represented by the $\bigotimes$ symbol.

It is important to understand that the sequence form specifies probabilities on sequences of actions *for a single player*. Thus, they are not the same as paths in the game tree; indeed, the sequence r^* for player 2 appears in two separate paths of the game tree, as player 2 has two nodes in the corresponding information set.

Say we have a set of probability distributions over actions at each information set, with σ_a denoting the probability of playing action a. We may construct a corresponding sequence-form strategy by applying the following equation in

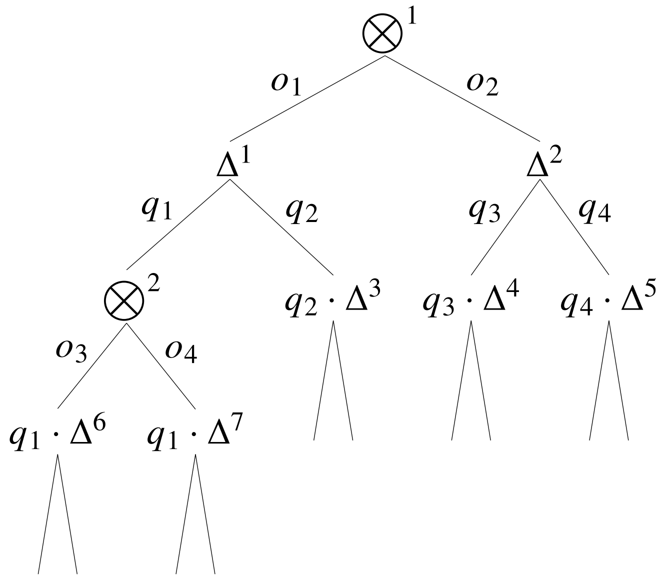

Figure 8.4 An example treeplex with 7 decision points (denoted by Δ) and two observation points (denoted by $\bigotimes$). Only internal edges are given labels, no labels are given to actions leading to leaf nodes. For each decision point, its scaling factor is listed, so e.g. $q_1 \cdot \Delta^6$ means that the simplex for the 6'th decision point is scaled by the value of q_1.

top-down fashion (so that x_{p_j} is always assigned before x_a):

$$x_a = x_{p_j}\sigma_a, \forall j \in \mathcal{J}, a \in A_j. \tag{8.3}$$

For a two-player zero-sum game, the payoff matrix A associated with the sequence-form setup is a sparse matrix, with each row corresponding to a sequence of the x player and each column corresponding to a sequence of the y player. Each leaf has a cell in A at the pair of sequences that are last visited by each player before reaching that leaf, and the value in the cell is the payoff to the maximizing player. Cells corresponding to pairs of sequences that are never the last pair of sequences visited before a leaf have a zero. If we let X be the set of sequence-form strategies for player 1 and Y be the set of sequence-form strategies for player 2, then the expected value under the pair of strategies $x \in X$, $y \in Y$ is $x^\top Ay$, which is bilinear. Thus, we have a bilinear saddle-point problem as desired:

$$\min_{x \in X} \max_{y \in Y} \langle x, Ay \rangle.$$

With this setup we now have an algorithm for computing a Nash equilibrium in a zero-sum EFG: Choose a distance-generating functions d_x, d_y, and run online mirror descent (OMD) for each player, using either of our folk-theorem setups from Chapter 6. However, this has one issue. Recall the update for OMD (also known as a *prox mapping* or *prox update*):

$$x_{t+1} = \operatorname*{arg\,min}_{x \in X} \langle \gamma g_t, x \rangle + D(x \| x_t),$$

where $D(x\|x_t) = d_x(x) - d_x(x_t) - \langle \nabla d_x(x_t), x - x_t \rangle$ is the Bregman divergence from x_t to x. In order to run OMD, we need to be able to compute this prox mapping. When X is a simplex the prox mapping is fairly straightforward to compute: the entropy DGF updates are closed-form, and updates for the Euclidean DGF can be computed in $n \log n$ time, where n is the number of actions. For treeplexes this question becomes more complicated.

In principle, we could use the standard Euclidean distance for d. In that case the update can be rewritten as

$$x_{t+1} = \operatorname*{arg\,min}_{x \in X} \|x - (x_t - \gamma g_t)\|_2^2,$$

which means that the update requires us to project onto a treeplex. This can be done in $n \cdot d \cdot \log n$ time, where n is the number of sequences and d is the depth of the decision space of the player. While this is acceptable, it turns out there are smarter ways to compute these updates which take linear time in n.

8.4 Dilated Distance-Generating Functions

We will see two ways to construct regret minimizers for treeplexes. The first is based on choosing an appropriate distance-generating function (DGF) for the treeplex, such that prox mappings are easy to compute. To that end, we now introduce what are called *dilated DGFs*. In dilated DGFs we assume that we have a DGF d_j for each information set $j \in \mathcal{J}$. For the polytope X we construct the DGF

$$d(x) = \sum_{j \in \mathcal{J}_1} \beta_j x_{p_j} d_j \left(\frac{x^j}{x_{p_j}} \right),$$

where $\beta_j > 0$ is the weight on information set j.

Dilated DGFs have the nice property that the proximal update can be computed recursively as long as we know how to compute the simplex update for each j. Let x^j, g_t^j, etc., denote the slice of a given vector corresponding to sequences belonging to information set j. The prox update is

$$\begin{aligned}
&\arg\min_{x \in X} \langle g_t, x\rangle + D(x\|x_t) \\
&= \arg\min_{x \in X} \langle g_t, x\rangle + d(x) - d(x_t) - \langle \nabla d(x_t), x - x_t\rangle \\
&= \arg\min_{x \in X} \langle g_t - \nabla d(x_t), x\rangle + d(x) \\
&= \arg\min_{x \in X} \sum_{j \in \mathcal{J}} \left(\langle g_t^j - \nabla d(x_t)^j, x^j\rangle + \beta_j x_{p_j} d_j(x^j / x_{p_j}) \right) \\
&= \arg\min_{x \in X} \sum_{j \in \mathcal{J}} x_{p_j} \left(\langle g_t^j - \nabla d(x_t)^j, x^j / x_{p_j}\rangle + \beta_j d_j(x^j / x_{p_j}) \right).
\end{aligned}$$

Now we may consider some information set j with no descendant information sets. Since x_{p_j} is on the outside of the parentheses, we can compute the update at j as if it were a simplex update, and the value at the information set can be added to the coefficient on x_{p_j}. That logic can then be applied recursively. Thus, we can traverse the treeplex in bottom-up order, and at each information set we can compute the value for x_{t+1}^j in however long it takes to compute an update for a simplex with DGF d_j.

If we use the entropy DGF for each $j \in \mathcal{J}$ and set the weights $\beta_j = 1$, then we get a DGF for X that is strongly convex modulus 1 with respect to a specialized *treeplex* ℓ_1 *norm* $\|\cdot\|_{X,1}$. One of the key aspects of the treeplex ℓ_1 norm is that it is a norm on the space of *leaves* in the treeplex. Measuring the size of a payoff vector only over the leaves makes sense in light of Eq. (8.2), which shows that every non-leaf entry in a sequence-form vector is linearly dependent on the leaf sequences. We let $\mathcal{E}_X$ be the set of leaf sequences for a treeplex X.

The treeplex ℓ_1 norm for a given treeplex X and a vector $g \in \mathbb{R}^{|\mathcal{E}_X|}$ is denoted as $\|g\|_{X,1} = \|g^{\hat{\emptyset}}\|_{X^{\hat{\emptyset}},1}$, and it is defined recursively in terms of treeplex ℓ_1 norms of subtreeplexes. To describe the norm, let $\mathcal{L}_X$ be an index set of leaves in treeplex X, $\mathcal{J}_X$ an index set of information sets in X, and O_X an index set of observation points in X. Then, the treeplex ℓ_1 can be described as taking absolute values at leaves, summing over children at information sets, and taking maximum over children at observation points. Formally, we get

$$\|g^j\|_{X^j,1} = \begin{cases} |g_j| & \text{if } j \in \mathcal{L}_X \\ \sum_{a \in A_j} \|g^{ja}\|_{X^{ja},1} & \text{if } j \in \mathcal{J}_X \\ \max_{o \in O_j} \|g^{jo}\|_{X^{jo},1} & \text{if } j \in O_X \end{cases}.$$

In the above, we use g^j to denote the slice of the vector g corresponding to a given decision point, observation point, or leaf j, and X^j to denote the corresponding subtreeplex rooted at j. We use the notation ja to denote the subsequent observation point after taking action at a at information set j, and similarly jo denotes the subsequent information set after seeing observation o at observation point j.

The dual norm of the treeplex ℓ_1 norm is the treeplex ℓ_∞ norm. This norm is also obtained recursively, but switches when sums and maxes are taken:

$$\|g^j\|_{X^j,\infty} = \begin{cases} |g_j| & \text{if } j \in \mathcal{L}_X \\ \max_{a \in A_j} \|g^{ja}\|_{X^{ja},\infty} & \text{if } j \in \mathcal{J}_X \\ \sum_{o \in O_j} \|g^{jo}\|_{X^{jo},\infty} & \text{if } j \in O_X \end{cases}.$$

When all payoffs are bounded to lie in $[0, 1]$, then it is possible to show that any feasible payoff vector g generated by choosing a strategy for the opposing player has dual norm $\|g\|_{X,\infty} \leq 1$. Secondly, the polytope diameter as measured by the dilated entropy DGF can be bounded by $\ln |\mathcal{V}_X|$, where $\mathcal{V}_X$ is the number of vertices of X. The number of vertices can be related to the normal-form representation of X; it is exactly the same as the number of strategies in the normal-form representation, after removing redundant normal-form actions.

If we instantiate the optimistic online mirror descent with the dilated entropy DGF for X and Y we get an algorithm that converges at a rate of

$$O\left(\frac{2 \ln |\mathcal{V}_X| + 2 \ln |\mathcal{V}_Y|}{T}\right),$$

where $\mathcal{V}_X, \mathcal{V}_Y$ is the number of vertices of each treeplex polytope. This gives the fastest theoretical rate of convergence among gradient-based methods. However, this only works for the optimistic and non-optimistic OMD algorithm. Regret matching and RM$^+$ were for simplex domains exclusively. Next we derive a way to use these locally at each information set. It turns out that faster practical performance can be obtained this way, though the theoretical rate of convergence is worse.

8.5 Counterfactual Regret Minimization

The framework for utilizing simplex regret minimizers at each information set is called *counterfactual regret minimization* (CFR). CFR is based on deriving an upper bound on regret, which allows decomposition into local regret minimization at each information set. We are interested in minimizing the standard regret notion over the sequence form:

$$R_T = \sum_{t\in[T]} \langle g_t, x_t\rangle - \min_{x\in X} \sum_{t\in[T]} \langle g_t, x\rangle.$$

To get the decomposition, we will define a local notion of regret which is defined with respect to behavioral strategies $\sigma \in \times_j \Delta^j =: \Sigma$ (we derive the decomposition for a single player, say player 1. Everything is analogous for player 2).

We saw in Section 8.3 that it is always possible to go from behavioral form to sequence form using the following recurrence, where assignment is performed in top-down order.

$$x_a = x_{p_j}\sigma_a, \forall j \in \mathcal{J}, a \in A_j. \tag{8.4}$$

It is also possible to go the other direction (though this direction is not a unique mapping, as one has a choice of how to assign behavioral probabilities at information sets j such that $x_{p_j} = 0$). These procedures produce payoff-equivalent strategies for perfect-recall EFGs.

For a behavioral strategy vector σ (or loss vector g_t) we say that σ^j is the slice of σ corresponding to information set j. $\sigma^{j\downarrow}$ is the slice corresponding to j, and every information set below j. Similarly, $\Sigma^{j\downarrow}$ is the set of all behavioral strategy assignments for the subset of simplexes that are in the tree of simplexes rooted at j.

We let C_{ja} be the set of next information sets belonging to player 1 that can be reached from j when taking action a. In other words, the set of information sets whose parent sequence is a.

Now, let the *value function* at time t for an information set j belonging to player 1 be defined as

$$V_t^j(\sigma) = \langle g_t^j, \sigma^j\rangle + \sum_{a\in A_j} \sum_{j'\in C_{ja}} \sigma_a V_t^{j'}(\sigma^{j'\downarrow}),$$

where $\sigma \in \Sigma^{j\downarrow}$. Intuitively, this value function represents the value that player 1 derives from information set j, assuming that i played to reach it, i.e. if we counterfactually set $x_{p_j} = 1$.

The *subtree regret* at a given information set j is

$$R_T^{j\downarrow} = \sum_{t\in[T]} V_t^j(\sigma_t^{j\downarrow}) - \min_{\sigma\in\Sigma^{j\downarrow}} \sum_{t\in[T]} V_t^j(\sigma),$$

Note that this regret is with respect to the behavioral form.

The local loss that we will eventually minimize is defined as

$$\hat{g}_{t,a}^j = g_{t,a} + \sum_{j'\in C_{ja}} V_t^{j'}(\sigma_t^{j'\downarrow}).$$

For each information set j, the loss will depend linearly on σ^j once we take the inner product $\langle \hat{g}_t^j, \sigma^j \rangle$; σ^j does *not* affect information sets below j, since we use σ_t in the value function for child information sets j'.

Now we show that the subtree regret decomposes in terms of local losses and subtree regrets.

Theorem 8.1 *For any $j \in \mathcal{J}$, the subtree regret at time T satisfies*

$$R_T^{j\downarrow} = \sum_{t\in[T]} \langle \hat{g}_t^j, \sigma_t^j \rangle - \min_{\sigma\in\Delta^j} \left(\sum_{t\in[T]} \langle \hat{g}_t^j, \sigma \rangle - \sum_{a\in A_j, j'\in C_{ja}} \sigma_a R_T^{j'\downarrow} \right).$$

Proof Using the definition of subtree regret we get

$$\begin{aligned} R_t^{j\downarrow} &= \sum_{t\in[T]} V_t^j(\sigma_t^{j\downarrow}) - \min_{\sigma\in\Sigma^{j\downarrow}} \left(\sum_{t\in[T]} \langle g_t^j, \sigma^j \rangle + \sum_{a\in A_j, j'\in C_{ja}} \sigma_a V_t^{j'}(\sigma^{j'\downarrow}) \right) \\ &= \sum_{t\in[T]} V_t^j(\sigma_t^{j\downarrow}) - \min_{\sigma\in\Delta^j} \left(\sum_{t\in[T]} \langle g_t^j, \sigma \rangle + \sum_{a\in A_j, j'\in C_{ja}} \sigma_a \min_{\hat{\sigma}\in\Sigma^{j'\downarrow}} V_t^{j'}(\hat{\sigma}^{j'\downarrow}) \right) \\ &= \sum_{t\in[T]} V_t^j(\sigma_t^{j\downarrow}) - \min_{\sigma\in\Delta^j} \left(\sum_{t\in[T]} \langle \hat{g}_t^j, \sigma \rangle - \sum_{a\in A_j, j'\in C_{ja}} \sigma_a R_T^{j'\downarrow} \right). \end{aligned}$$

The first equality is by expanding $V_t^j(\sigma^{j\downarrow})$. The second equality is by minimizing sequentially. The third equality is by adding and subtracting $\sigma_a V_t^{j'}(\sigma_t^{j'\downarrow})$ for each $a \in A_j$, $j' \in C_{ja}$ and using the definition of $\hat{g}_t$ and $R_T^{j'\downarrow}$. The theorem follows, since $V_t^j(\sigma_t^{j\downarrow}) = \langle \hat{g}_t^j, \sigma_t^j \rangle$. □

The local regret that we will be minimizing at a given information set j is

$$\hat{R}_T^j := \sum_{t\in[T]} \langle \hat{g}_t^j, \sigma_t^j \rangle - \min_{\sigma\in\Delta^j} \sum_{t\in[T]} \langle \hat{g}_t^j, \sigma \rangle.$$

Note that this regret is in the behavioral form, and it corresponds exactly to the regret associated to locally minimizing $\hat{g}_t^j$ at each simplex j.

The CFR framework is based on the following theorem, which says that the sequence-form regret can be upper-bounded by the behavioral-form local regrets.

Theorem 8.2 *The regret at time T satisfies*

$$R_T = R_T^{\hat{0}\downarrow} \leq \max_{x\in X} \sum_{j\in\mathcal{J}} x_{p_j} \hat{R}_T^j,$$

where $\hat{\emptyset}$ is the root information set.[1]

Proof For the equality, consider the regret R_T over the sequence form polytope X. Since each sequence-form strategy has a payoff equivalent behavioral strategy in Σ and vice versa, we get that the regret R_T is equal to $R_T^{\hat{\emptyset}\downarrow}$ for the root information set.

By Theorem 8.1 we have for any $j \in \mathcal{J}$

$$\begin{aligned} R_T^{j\downarrow} &= \sum_{t\in[T]} \langle \hat{g}_t^j, \sigma_t^j \rangle - \min_{\sigma\in\Delta^j} \left(\sum_{t\in[T]} \langle \hat{g}_t^j, \sigma \rangle - \sum_{a\in A_j, j'\in C_{ja}} \sigma_a R_T^{j'\downarrow} \right) \\ &\leq \sum_{t\in[T]} \langle \hat{g}_t^j, \sigma_t^j \rangle - \min_{\sigma\in\Delta^j} \sum_{t\in[T]} \langle \hat{g}_t^j, \sigma \rangle + \max_{\sigma\in\Delta^j} \sum_{a\in A_j, j'\in C_{ja}} \sigma_a R_T^{j'\downarrow}, \end{aligned} \tag{8.5}$$

where the inequality is by the fact that independently minimizing the terms $\sum_{t\in[T]} \langle \hat{g}_t^j, \sigma \rangle$ and $-\sum_{a\in A_j, j'\in C_{ja}} \sigma_a R_T^{j'\downarrow}$ is smaller than jointly minimizing them.

Now we may apply (8.5) recursively in top-down fashion starting at $\hat{\emptyset}$ to get the theorem. □

A direct corollary of Theorem 8.2 is that if the counterfactual regret at each information set grows sublinearly then overall regret grows sublinearly. This is the foundation of the *counterfactual regret minimization* (CFR) framework for minimizing regret over treeplexes. The CFR framework can succinctly be described as

(i) Instantiate a local regret minimizer for each information set simplex Δ^j.
(ii) At iteration t, for each $j \in \mathcal{J}$, feed the local regret minimizer the counterfactual regret $\hat{g}_t^j$.
(iii) Generate x_{t+1} as follows: ask for the next recommendation from each local regret minimizer. This yields a set of simplex strategies, one for each information set. Construct x_{t+1} via (8.4).

Thus, we get an algorithm for minimizing regret on treeplexes based on minimizing counterfactual regrets. In order to construct an algorithm for computing a two-player zero-sum Nash equilibrium based on a CFR setup, we may invoke the folk theorem from Chapter 6 (or a variation) using the sequence-form strategies generated by CFR. Doing this yields an algorithm that converges to a Nash equilibrium a rate of $O\left(\frac{1}{\sqrt{T}}\right)$

While CFR is technically a framework for constructing local regret minimizers, the term "CFR" is often overloaded to mean the algorithm that results from

[1] If there is more than one root information set then we can add a dummy single information set that precedes all the root information sets.

using the folk theorem with uniform averages, and using regret matching as the local regret minimizer at each information set. CFR$^+$ is the algorithm resulting from using the alternation setup, taking linear averages of strategies, and using RM$^+$ as the local regret minimizer at each information set.

Algorithm 1 CFR(RM$^+$)($\mathcal{J}, X$)

Input: $\mathcal{J}$: set of infosets, $X \in \mathbb{R}^n_{\geq 0}$: sequence-form strategy space
procedure SETUP
 $\mathbf{Q} = \text{zeros}(n)$; the all-zero vector
 $t = 1$

function NEXTSTRATEGY
 $x = 0 \in \mathbb{R}^n$
 $x_\emptyset = 1$
 for all $j \in \mathcal{J}$ in *top-down* order **do**
  $s = \sum_{a \in A_j} \mathbf{Q}_a$
  for all $a \in A_j$ **do**
   if $s = 0$ **then**
    $x_a = x_{p_j} / |A_j|$
   else
    $x_a = x_{p_j} \times \mathbf{Q}_a / s$
 return x

function OBSERVELOSS($g_t \in \mathbb{R}^n$)
 for all $j \in \mathcal{J}$ in *bottom-up* order **do**
  $s = \sum_{a \in A_j} \mathbf{Q}_a$
  $v = 0$ ▷ value of infoset j
  if $s = 0$ **then**
   $v = \sum_{a \in A_j} g_{t,a} / |A_j|$
  else
   $v = \sum_{a \in A_j} \langle g_{t,a}, \mathbf{Q}_a / s \rangle$
  $g_{t,p_j} = g_{t,p_j} + v$ ▷ construct local loss $\hat{g}_t$
  for all $a \in A_j$ **do**
   $\mathbf{Q}_a = [\mathbf{Q}_a + (v - g_{t,a})]^+$
 $t = t + 1$

We now show pseudocode for implementing the CFR algorithm with the RM$^+$ regret minimizer. In order to compute Nash equilibria with this method one would use CFR as the regret minimizer in one of the folk-theorem setups from Chapter 6. NextStrategy implements the top-down recursion in (8.4), while computing the update corresponding to RM$^+$ at each j. ObserveLoss uses bottom-up recursion to keep track of the regret-like sequence Q_a, which is based on $\hat{g}_{t,a}$ in CFR.

The pseudocode assumes that there is a dummy *empty sequence* $\emptyset$ at the root of the treeplex with no corresponding j (this corresponds to a single-action dummy information set at the root, but leaving out that dummy information set in the index set $\mathcal{J}$). This is similar, but distinct from, the root *infoset* used in Theorem 8.2. This makes code much cleaner because there is no need to worry about the special case where a given j has no parent sequence, at the low cost of increasing the length of each player's sequence-form vector by 1.

8.6 Numerical Comparison of CFR methods and OMD-like methods

Figure 8.5 shows the performance of three CFR variants and two optimistic OMD-based algorithms with dilated entropy (DOMD(Entropy)) and dilated Euclidean (DOMD(Euclidean)) DGFs for solving three EFGs: *Kuhn poker* and *Leduc poker*, simplified poker games that are standard in EFG solving, and *Sheriff*, a simplified version of the Sheriff of Nottingham game. The experiments were run with the LiteEFG library (Liu et al., 2024).

All algorithms use alternation, and all algorithms were run for $10,000$ iterations. An algorithm uses uniform averaging unless 'lin' is appended to the name, in which case it uses linear averaging. The stepsizes for the optimistic OMD algorithms were lightly tuned (I tried about 5 stepsizes for each, and picked the one with the lowest saddle-point residual after the $10,000$ iterations).

Similar to the case of two-player zero-sum normal-form games in Chapter 6, we see that CFR$^+$ performs very well with linear averaging, especially in Leduc poker. Optimistic OMD with the dilated Euclidean DGF also performs extremely well in Kuhn poker and Sheriff. This is a common occurrence in the literature, where CFR$^+$ (with linear averaging) is usually the best algorithm for solving poker games, whereas other methods often perform better for non-poker games. Kuhn poker breaks with this categorization, which is also consistent in the literature. Most likely, this is because Kuhn poker is too "simple" of a poker game.

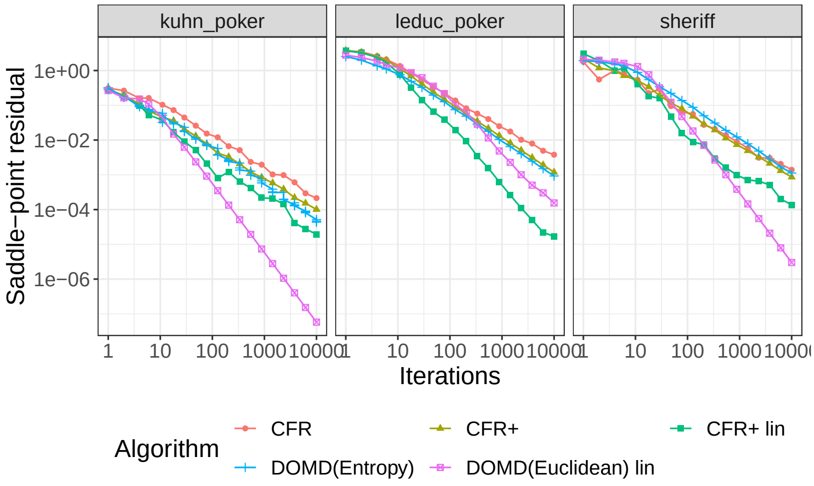

Figure 8.5 Solution accuracy as a function of the number of algorithm iterations in three EFGs: Kuhn poker (left), Leduc poker (center), and Sheriff (right). Results are shown for CFR with regret matching, CFR^+ with uniform averaging (labeled as CFR+), CFR^+ with linear averaging (labeled as "CFR+ lin"), optimistic OMD with the dilated entropy DGF, and optimistic OMD with the dilated Euclidean DGF and linear averaging. Both axes are shown on a log scale.

8.7 Stochastic Gradient Estimates

So far we have operated under the assumption that we can easily compute the gradients for each player, i.e. the matrix-vector products Ay_t and $-A^\top x_t$, for the EFG that we are trying to solve. While these can indeed be computed in time linear in the size of the game tree, there are cases (such as in poker AI) where the game tree is so large that even one traversal is too much. In that case, we are interested in developing methods that can work with some stochastic gradient estimator (say for the x player) $\tilde{g}_t$ of the gradient. Typically, one would consider unbiased gradient estimators, i.e. $\mathbb{E}[\tilde{g}_t] = Ay_t$.

Assuming that we have a gradient estimator $\tilde{g}_t$ for each time t, a natural approach for attempting to compute a solution would be to apply our previous approach of running a regret minimizer for each player and using the folk theorem, but now using $\tilde{g}_t$ at each iteration, rather than g_t. If our unbiased gradient estimator $\tilde{g}_t$ is reasonably accurate then we might expect that this approach should still yield an algorithm for computing a Nash equilibrium. This turns out to be the case.

Theorem 8.3 *Assume that each player uses a bounded unbiased gradient* *estimator for their loss at each iteration. Then for all* $p \in (0,1)$*, with probability at least* $1 - 2p$

$$\xi(\bar{x}, \bar{y}) \leq \frac{\tilde{R}_T^1 + \tilde{R}_T^2}{T} + \left(2\Delta + \tilde{M}_1 + \tilde{M}_2\right)\sqrt{\frac{2}{T}\log\frac{1}{p}},$$

where $\tilde{R}_T^i$ *is the regret incurred under the losses* $\tilde{g}_t^i$ *for player* i*,* $\Delta = \max_{z,z' \in Z} u_2(z) - u_2(z')$ *is the* payoff range *of the game, and* $\tilde{M}_1 \geq \max_{x,x' \in X} \langle \tilde{g}_t, x - x' \rangle, \forall \tilde{g}_t$ *is a bound on the "size" of the gradient estimate, with* M_2 *defined analogously.*

We will not show the proof here, but it follows straightforwardly from introducing the discrete-time stochastic process

$$d_t := g_t(x_t - x)) - \tilde{g}_t(x_t - x),$$

observing that it is a martingale difference sequence, and applying the Azuma-Hoeffding inequality (see Theorem B.1).

With Theorem 8.3 in hand, we just need a good way to construct gradient estimates $\tilde{g}_t \approx Ay_t$. Generally, one can construct a wide array of gradient estimators by using the fact that Ay_t can be computed by traversing the EFG game tree: at each leaf node z in the tree, we add $-u_1(z)y_a$ to $g_{t,a'}$, where a is the last sequence taken by the y player, and a' is the last sequence taken by the x player. To construct an estimator, we may choose to sample actions at some subset of nodes in the game tree, and then only traverse the sampled branches, while taking care to normalize the eventual payoff so that we maintain an unbiased estimator. One of the most successful estimators construct this way is the *external sampling* estimator. In external sampling when computing the gradient Ay_t, we sample a single action at every node belonging to the y player or chance, while traversing all branches at nodes belonging to the x player.

Figure 8.6 shows the performance when using external sampling in CFR (CFR with sampling is usually called Monte-Carlo CFR or MCCFR), CFR$^+$, and optimistic OMD. Performance is shown on Kuhn poker, Leduc poker, and Sheriff again. In the deterministic case we saw that CFR$^+$ was much faster than CFR, and also faster than the theoretically-superior optimistic OMD in Leduc poker. In the stochastic case the results are similar for all the algorithms, and in fact CFR performs slightly better than CFR$^+$. OMD with the dilated entropy DGF, previously the worst algorithm in the deterministic case, now performs better slightly better than the other algorithms, though only marginally so.

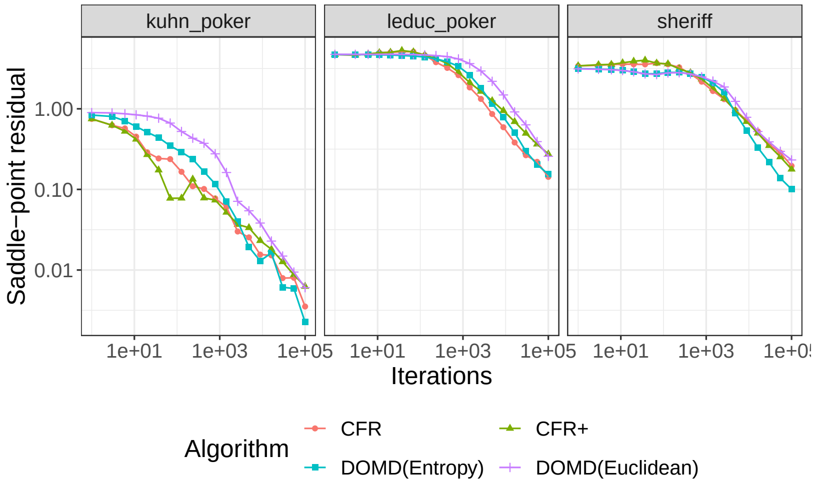

Figure 8.6 Performance of CFR, CFR$^+$, and optimistic OMD (with both the dilated entropy and Euclidean DGFs) when using the external sampling gradient estimator.

8.8 Search in Extensive-Form Games

We previously saw how to compute a Nash equilibrium of a two-player zero-sum extensive-form game (EFG) by using dilated distance-generating functions or the CFR framework. We also saw that even if computing gradients $g_t = Ay_t$ is too time-consuming we can still run algorithms using gradient estimates constructed via sampling. However, for some real-world games such as two-player no-limit Texas hold'em, this is still not enough. The game tree in this game has roughly 10^{170} nodes, and the strategy space is much too large to even write down strategy iterates. Faced with this situation, we need to make even coarser approximations to our problem.

One major innovation for solving large-scale poker games was the use of *real-time search*. Traditionally, poker AIs were created by precomputing an approximate Nash equilibrium for some extremely coarsened representation of the full game using e.g. CFR$^+$. Then, that offline strategy was simply employed during play. In real-time search, the precomputed Nash equilibrium approximation is refined in real time for subgames encountered during live play. This allows the AI to reason in much more detail, especially towards the end of the game, where the encountered subtree is manageable in size. In order to understand how search works in EFGs, we will first show how it works in the simpler setting of perfect-information EFGs, where there are no information sets, and so players know exactly which node they are currently at. Search in perfect-information EFGs has historically been extremely successful, it was used in AI milestones on Chess and Go.

8.8.1 Backward Induction

Perfect-information EFGs (meaning that all information sets consist of a single node) can be solved via backward induction, as discussed in Section 8.1. Since the game is played on a tree, and a player always knows exactly where in the tree they are, we can reason about the optimal strategy at a given node purely by considering the subgame rooted at the node. We do not need to worry about what happens in any other parts of the game tree. Backward induction exploits this fact by recursively solving every subgame. It starts at leaf nodes, and then at any internal node, the algorithm pick the action that leads to the best subgame for the player acting at the node (breaking ties arbitrarily). Similarly, search methods can exploit this structure.

8.8.2 Search in Games

In search, we search for a solution in real-time during play. Say that we are playing chess, which is a perfect-information EFG. Say that some set of moves already happened, resulting in the board state shown below:

In order to decide a next move for black, we can now perform real-time search. We perform backward induction starting at the subgame rooted at the current board state. What this means is that we try all sequences of legal moves starting with the current state, and then we pick the best action based on having solved the subgame via backward induction. However, unless we are close to the end of the game, the size of the subgame usually makes backward induction much too slow. Instead, the search is performed only up to a certain depth, say 10 moves ahead. This generally won't get us to a leaf node of the game, and so instead we replace the nodes at depth 10 with fake leaf nodes that we assign some heuristic estimate of the unique value that would have resulted from backward induction (we will call these fake leaf nodes *subgame leaf nodes*). In order to do that, we need to construct an estimate of what value an internal node would have in the solution. A visualization is shown in Figure 8.8.

In order to estimate the value of some internal node h in the game tree, we assume that we have some *value estimator* $v : H' \to \mathbb{R}$, where H' is the set of nodes in the game tree that are leaf nodes in the subgame. Each subgame leaf node h is then assigned the value $v(h)$ in the subgame. In perfect-information games each node h has some unique value associated to the solution arising

1 e4 e5 2 ♘f3 ♘c6 3 ♗b5

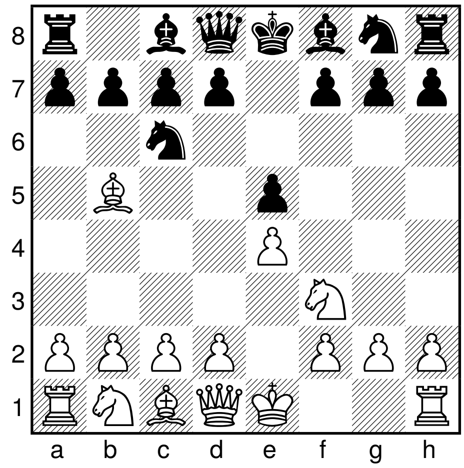

Figure 8.7 A chess board.

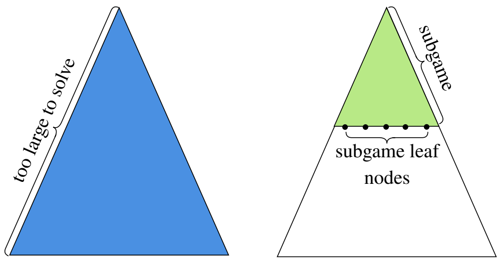

Figure 8.8 A large game truncated to a depth-limited subgame starting at the root.

from backward induction. In that case, our goal is simply to have $v(h)$ be a good approximation to this unique value. If $v(h)$ provides perfect estimates then backward induction in the subgame recovers the solution to the original game.

So how do you get a value estimator? It can be handcrafted based on domain knowledge (this was done for *Deep Blue*, a chess AI which beat Garry Kasparov, at the time considered the best chess player in the world); it can be learned by training on expert human games (this was done by *AlphaGo*, a Go AI which beat Lee Sidol, a top-tier professional Go player); or finally it can be done via self-play (this was done by *AlphaZero*, a generalization of AlphaGo, and *Pluribus*, a poker AI that beat humans at 6-player poker).

For imperfect-information games such as poker, things are more complicated. The primary issue is that backward induction no longer works: The value of a given node cannot be understood purely in terms of the subtree rooted at the node. Instead, we must take into account the rest of the game tree. Further complicating matters is the fact that a node does not have a single well-defined value; the value of a node may change depending on which Nash equilibrium we are considering. Finally, even if we manage to estimate the value of a node in equilibrium, we may end up choosing a strategy where the opponent can best respond in order to exploit us in the truncated part of the game tree. This is easily seen by considering the EFG representation of rock-paper-scissors: At the root node player 1 chooses rock, paper, or scissors. Then, player 2 has a single information set containing all three nodes corresponding to each choice for player 1, and they choose rock, paper, or scissors at that information set. If we truncate the game at depth 1 and assign each player 2 node its value in equilibrium (which is 0), then player 1 ends up with 3 actions, all leading to a payoff of 0. Thus, for the subgame player 1 can choose any pure strategy, e.g. always play rock, and based on the subgame think that they achieve a value of zero. However, once we play in real time, if our opponent knows that we truncated the game and picked rock, they may exploit us by playing paper.

The above issue arises from a loss of contextual information: the value of the node corresponding to player 1 choosing rock is affected by how frequently they choose paper and scissors as well. We can add this contextual information to our value predictor as follows: instead of deterministically predicting a node value $v(h)$, we make our node-value prediction contextual by conditioning on p. Then, $v(h|p)$ is our estimate of the value for h, conditional on a probability distribution p over nodes at the truncation level. For the rock-paper-scissors example, $v(h_r|[p_r, p_p, p_s])$ would estimate the value of, say, the rock node h_r conditional on the distribution $[p_r, p_p, p_s]$ over the 3 possible nodes. This leads to a much more complicated value estimator, since we are now trying to construct a mapping from $H' \times \Delta^{|H'|}$ to $\mathbb{R}$. This is the approach taken in the *DeepStack* poker AI, which beat a group of professional poker players in two-player no-limit Texas hold'em. Values are estimated using a deep neural network that was pretrained by generating random distributions over subgame leaf nodes, and then solving each of the subgames defined by truncating the *top* of the game, and having a chance root node that randomizes over the subgame leaf nodes using the randomly generated distribution.

Once an estimator v has been constructed, real-time search with this setup works as follows:

(i) Define a subgame by looking k moves ahead.
(ii) Solve the subgame using a regret-minimization algorithm for EFGs (e.g. CFR$^+$).
(iii) On each iteration t of the regret-minimization algorithm:
   (a) Algorithms chooses subgame strategies x_t, y_t.
   (b) x_t, y_t defines a probability distribution $p(Z)$ over subgame leaf nodes.
   (c) For each leaf node z, ask value predictor for estimates $v(z, p(Z))$.
   (d) Set loss for the x player to $g_t = A_t y_t$, where A_t is the payoff matrix associated to the subgame with subgame leaf estimates $v(z, p(Z))$. Define the loss for the y player analogously.

If the value network is perfect, then this setup computes a strategy for the subgame that is part of some Nash equilibrium in the full game.

To summarize the approach described here: we train our value network offline, e.g. by generating random distributions over nodes, and solving those subgames. This generates training data. Then, during play we use the already-trained value network to solve subgames as we encounter them.

This still leaves open the question of how to solve the subgames needed to create the value network, since those subgames could be very large themselves (e.g. subgames starting near the root of the game tree). One way to do it is to start by randomly generating shallow games near the bottom of the game, say depth d_1. Once we have a good value network for predicting the value of nodes at depth d_1, we can move up one level. Next, we randomly generate distributions over nodes at depth d_2, and truncate those games at depth d_1 using our value network that we already constructed for depth d_1. We can then apply this recursively.

So far we have described our methodology and examples as if we are solving a depth-limited subgame starting at the root node of the game tree. However, in practice we would like to solve subgames starting at arbitrary decision points in the game. In perfect-information games this is easily done. We may treat it exactly the same as solving from the root, since every node provides a well-defined subgame with that node as the root. However, in imperfect-information games this is not so.

To construct an imperfect-information EFG subgame, we assume that we have so far been playing according to some *blueprint* strategy which we computed ahead of time (our opponent need not follow the blueprint strategy in practice). Typically, this blueprint strategy would be computed using CFR$^+$ on a very coarsened abstraction of the game.

When constructing a non-root subgame in an imperfect-information game, we will in general not know exactly which node we are at, and so instead we would have to start the subgame at the information set that we are currently at. But even taking all the nodes in the current information set as the root (and applying Bayes' rule to derive a chance node that selects among them), will not be enough. In particular, nodes in subtrees rooted at the information set may be in information sets that contain nodes that are *not* in any of the subtrees. To remedy this, we construct our subgame by starting with all nodes in subtrees rooted at the current information set. Then, we add to our subgame every node that shares an information set with at least one node currently in our subgame. We then repeatedly add nodes in this fashion, until we reach the point where there are *no* nodes outside the subgame which share an information set with any node in our (now much larger) subgame. Finally, in order to finish our construction we need a probability distribution over all the nodes that are at the top level (i.e. same level as the information set we wanted to create a subgame for) of this new subgame. The most naive approach would be to make a single chance node as the root, and use the conditional distribution over the set of top-level nodes given our blueprint strategy. This approach is typically called *unsafe subgame solving*. The reason it is called unsafe is that we are generally not guaranteed that we will be weakly better off by applying subgame solving, as compared to our blueprint strategy. By not considering the rest of the game, it turns out that we might open ourselves up to exploitation. Nonetheless, unsafe subgame solving is often used in practice.

There are various methods for performing "safe" subgame solving. These typically require adding additional gadgets to the unsafe subgame construction, either by enforcing that the opponent achieves a certain level of utility in the subgame (this prevents us from overfitting to the subgame), or replacing the initial chance node with a number of opponent nodes, where they can reject the subgame unless they achieve a certain utility level.

8.9 Historical Notes

The sequence form was discovered in the USSR in the 60s (Romanovskii, 1962) and later rediscovered independently (von Stengel, 1996; Koller et al., 1996). Dilated DGFs for EFGs were introduced by Hoda et al. (2010). Hoda et al. (2010) proved that any dilated DGF constructed from strongly-convex simplex DGFs must also be strongly convex, though they did not derive explicit bounds. Deriving an explicit strong convexity modulus for the dilated entropy DGF was studied by several papers (Kroer et al., 2020; Farina et al., 2021a). The final version that we present here was given by Fan et al. (2024), which also introduced the specialized treeplex ℓ_1 and ℓ_∞ norms. Fan et al. (2024)

also show corresponding lower bounds which imply that the bound presented in Section 8.4 is tight up to log factors. An explicit bound for the dilated Euclidean DGF can be found in Farina et al. (2019c), though it is possible that their bound can be improved.

CFR-based algorithms were used as the algorithm for computing Nash equilibrium in all the recent milestones where AIs beat human players at various poker games (Bowling et al., 2015; Moravčík et al., 2017; Brown and Sandholm, 2018, 2019b). CFR was introduced by Zinkevich et al. (2007). Many later variations have been developed, for example the stochastic method MC-CFR (Lanctot et al., 2009), and variations on which local regret minimizer to use in order to speed up practical performance (Tammelin et al., 2015; Brown and Sandholm, 2019a). The proof of CFR given here is a simplified version of the more general theorem developed in Farina et al. (2019b). The plots on CFR vs EGT are from Kroer et al. (2020).

The bound on error from using a stochastic method in Theorem 8.3 is from Farina et al. (2020), and the plots on stochastic methods are from that same paper. External sampling and several other EFG gradient estimators were introduced by Lanctot et al. (2009).

Search was used in several poker AIs that beat human poker players of various degrees of expertise, both in two-player poker (Moravčík et al., 2017; Brown and Sandholm, 2018) and 6-player poker (Brown and Sandholm, 2019b). *Endgame solving*, where we solve the remainder of the game, was studied in the unsafe version by Ganzfried and Sandholm (2015). Safe endgame solving was studied by Burch et al. (2014); Moravcik et al. (2016) and Brown and Sandholm (2017). The more general version of subgame solving where we do not have to solve to the end of the game was studied by Moravčík et al. (2017); Brown et al. (2018b,a).

The benchmark EFGs that we used were introduced by various others. Kuhn poker was designed by Harold W. Kuhn in 1950 (Kuhn, 2016). Leduc poker was introduced by Southey et al. (2005). Sheriff was introduced by Farina et al. (2019a).

Further reading.

An economics-focused introduction to EFGs can be found in Fudenberg and Tirole (1991). For the sequence-form linear-programming approach to computing Nash equilibria in EFGs, the chapter by Bernhard von Stengel in Nisan et al. (2007) is a good source, as well as Shoham and Leyton-Brown (2008). For CFR, I am not aware of a very intuitive coverage. I am partial to the regret-minimization perspective that we developed in Farina et al. (2019b). For a more "traditional" introduction to CFR, I recommend the survey by Neller and

Lanctot (2013). Regarding dilated distance-generating functions, the original paper by Hoda et al. (2010) is a good starting point. For the strongest results specifically on the dilated entropy, see Farina et al. (2025) and Fan et al. (2024). Search in imperfect-information games is a very recent topic, and there are no textbooks covering it. The references listed above are the best source for further reading.

9
Stackelberg equilibrium and Security Games

In this chapter we introduce *Stackelberg equilibrium*. Stackelberg equilibrium is an equilibrium notion for two-player general-sum games where one player is a *leader* and the other player is a *follower* (it can also be generalized to multiple leaders and/or followers). This model is appropriate for example when modeling competing firms and first-mover advantage or, as we will see, security settings centered around asset protection.

9.1 Stackelberg Equilibrium

We will consider a two-player normal-form game where there is a leader ℓ and a follower f. The leader has a finite set of actions A_ℓ and the follower has a finite set of actions A_f. We let Δ^ℓ, Δ^f denote the set of probability distributions over the leader and follower actions. We will consider a general-sum game with utilities $u_i(a_\ell, a_f)$ for $i \in \{\ell, f\}$. Given probability distributions $x \in \Delta^\ell, y \in \Delta^f$ over A_ℓ and A_f, we abuse notation slightly and let

$$u_i(x, y) = \mathbb{E}_{a_\ell \sim x, a_f \sim y} \left[u_i(a_\ell, a_f) \right],$$

We assume that the leader is able to *commit* to a strategy $x \in X$. Given x, the follower observes x and chooses their strategy from the best-response set

$$BR(x) = \arg\max_{y \in \Delta^f} u_f(x, y).$$

The goal of the leader is to choose a strategy x maximizing their utility subject to the follower best responding. Formally, they wish to solve

$$\max_{x \in \Delta^\ell} u_\ell(x, y) \text{ s.t. } y \in BR(x). \tag{9.1}$$

However, this optimization problem has a problem currently. Can you see what it is?

The issue is that $BR(x)$ may be set valued, and $u_\ell(x, y)$ generally would differ defending on which $y \in BR(x)$ is chosen. Because of this, we need a rule for how to choose among the best responses. In a *strong Stackelberg equilibrium* (SSE) we assume that the follower breaks ties in favor of the leader. In that case the optimization problem is

$$\max_{x \in \Delta^\ell, y \in BR(x)} u_\ell(x, y). \tag{9.2}$$

SSE is, in a sense, the most optimistic variant. Conversely, we may consider the most pessimistic assumption, that ties are broken adversarially. This yields the *weak Stackelberg equilibrium* (WSE)

$$\max_{x \in \Delta^\ell} \min_{y \in BR(x)} u_\ell(x, y). \tag{9.3}$$

Do not be misled by the minimax structure in this problem; it is not well-behaved the same way that our other minimax problems have been. This is because the feasible set for y depends on x, which introduces problems such as nonexistence (an example is give below for an inspection game). In practice SSE has been by far the most popular. One major advantage of SSE is that it is always guaranteed to exist, whereas WSE is not.

A first question we might ask ourselves is whether it always helps or hurts to be able to first commit to a strategy, as compared to playing a Nash equilibrium.

First, let us consider the zero-sum case. If we are in a zero-sum game, then we already saw from von Neumann's minimax theorem that we can represent the Nash equilibrium problem as

$$\min_{x \in \Delta^\ell} \max_{y \in \Delta^f} \langle x, Ay \rangle = \max_{y \in \Delta^f} \min_{x \in \Delta^\ell} \langle x, Ay \rangle.$$

It follows that Nash equilibrium and Stackelberg equilibrium are equivalent in this setting (since being able to do better than a Nash equilibrium would violate the minimax theorem, and we can always commit to a minimax strategy).

Second, consider the case where we restrict the leader to only committing to *pure* actions $a \in A_\ell$. Then it is easy to see that committing to a strategy first may hurt the leader (consider Rock, Paper, Scissors).

Finally, consider the case where we allow commitment a mixed strategy $x \in \Delta^\ell$. In this case it turns out that committing to a strategy helps the leader.

Theorem 9.1 *In a general-sum game, the leader achieves weakly more utility in SSE than in any Nash equilibrium.*

	cheat	no cheat
inspect	-6, -9	-1,0
no inspection	-10, 1	0, 0

Table 9.1 *The payoff matrix for an inspection game.*

Proof Consider the Nash equilibrium (x, y) that yields the highest utility for the leader. Suppose the leader commits to x. Since the follower breaks ties in favor of the leader, we get that if the leader commits to x then the follower can at worst pick y from $BR(x)$. If they don't pick y, then they must pick something that yields even better utility for the leader. So, by committing to x, the leader weakly improves on the Nash equilibrium. Thus, they do at least as well, since they can choose x, or something better. □

Similarly, it can be shown that the WSE solution is at least as good as *some* Nash equilibrium payoff for the leader (see von Stengel and Zamir (2010) for a proof). Thus, if we consider the range of payoffs $[L, H]$ from the lowest to highest in Stackelberg equilibrium, then that range lies above the range that we would get for Nash equilibrium.

A classic example of the difference between Nash equilibrium and Stackelberg equilibrium is in the context of *inspection games*. In an inspection game, an inspector chooses whether to inspect or not, and the inspectee chooses whether to cheat or not. An example game is shown below

The goal of the inspector is to deter cheating, and inspecting incurs a cost of -1. When cheating occurs the inspector incurs a heavy negative cost, whether detected or not (so the goal is not to catch cheaters, but rather to deter cheating). The inspectee gains utility from cheating undetected (-10, 1), but incurs a heavy fine if they cheat and are inspected (-6,-9).

There is a single unique Nash equilibrium in this game, where the inspector inspects with probability $\frac{1}{10}$, and the inspectee cheats with probability $\frac{1}{5}$. This yields expected utilities of (-2, 0) for the two players.

Now consider the same game, but where we allow the inspector to be the leader in a Stackelberg game. Any strategy that inspects with probability at least $\frac{1}{10}$ will make not cheating a best response for the follower. The SSE of the game is for the inspector to inspect with probability $\frac{1}{10}$ and the inspectee to not cheat. This yields expected utilities $(-\frac{1}{10}, 0)$, which is much better for the inspector. Note furthermore that if we consider the WSE solution concept, then the inspector must inspect with probability *strictly* greater than $\frac{1}{10}$ in order to make not cheating the only best response. But this means that a WSE does not exist, since for every leader strategy that inspects with probability $p > \frac{1}{10}$, the leader can improve their utility by inspecting with any probability in the open interval $(\frac{1}{10}, p)$.

In the normal-form game setup given above, an SSE can be computed in polynomial time. In particular, say that we want to maximize leader utility, subject to making a particular follower action $a_f \in A_f$ a best response. We may solve this problem using the following LP:

$$\max_{x \in \Delta^\ell} \sum_{a \in A_\ell} x_a u_\ell(a, a_f)$$
$$s.t. \sum_{a \in A_\ell} x_a u_f(a, a_f) \geq \sum_{a \in A_\ell} x_a u_f(a, a'_f), \ \forall a'_f \in A_f.$$

Now, in order to find the optimal strategy to commit to, we may iterate over all $a_f \in A_f$, solve the LP for each, and pick the optimal solution x^* associated to the LP with the highest value. This works because of the assumption that ties are broken in favor of the leader (convince yourself why).

Once we have the optimal strategy x^*, we may find the associated follower strategy simply by picking the pure strategy a_f for which x^* was the LP solution. Generally, it is easy to see that it is always enough to consider only pure strategies when choosing the follower strategy in an SSE (why?). The same holds true for WSE.

This LP-based algorithm also proves that an SSE is always guaranteed to exist.

9.2 Security Games

Stackelberg equilibrium models have been deployed extensively in asset protection scenarios, such as deployment of patrols in airports, poaching deterrence, coast guard patrolling, subway fare inspection, and others. This broad class of models are called *security games*.

In the security games model (SGM) a defender (the leader) is interested in protecting a set of targets using limited resources, while an attacker (the follower) is able to observe the strategy of the leader, and best respond to it. A classical example would be that of protecting an airport: say we have 5 vulnerable locations at the airport, but only 2 patrol units. How can we schedule the patrols to provide maximum coverage across the 5 vulnerable locations, while taking into account the fact that an attacker prefers certain locations to others?

The basic security games model has a set T of targets (note that we could have a single *physical* target appear twice in T, representing attacking that target, say, in the morning or evening). The defender controls a set of resources R that can be assigned to a *schedule* from a set $S \subseteq 2^T$ of possible schedules. A schedule is a subset of targets that are simultaneously covered if a resource is assigned that given schedule (for example in the airport example, a resource would be a patrol, and schedules would be the set of feasible patrols across targets). We say that a target is "covered" if the defender assigns a resource to a schedule that covers it. The action space for the attacker consists of choosing which single target to attack. In the basic SSG model, the utility function of both the defender and attacker depends only on which target is attacked, and whether it is covered or not. Formally, we say that the defender receives utility $u_d^c(t)$ if target t is attacked and covered, and utility $u_d^u(t)$ if target t is attacked and not covered. Similarly, the attacker gains utility $u_a^c(t)$ if target t is attacked and covered, and $u_a^u(t)$ if target t is attacked and not covered. If the resources R are not homogenous then there may be an *assignment function* $A : R \to S$ denoting the set of schedules s that resource r can be assigned.

For security games we will restrict our attention to SSE. Given a strategy x for the defender, we get a deployment of resources to targets for the defender, with an induced probability distribution $p_c(t|x)$ of whether each target is covered. A strategy for the attacker simply specifies a single target t to attack (recall that it is enough to consider pure strategies for the follower in Stackelberg equilibrium). For a strategy pair x, t the expected utility for the defender is $p_c(t|x)u_d^c(t) + (1 - p_c(t|x))u_d^u(t)$, with attacker utility defined analogously.

9.2.1 Algorithms for Security Games

So now that we have a game model for security games, can we just apply our LP result on computing SSE in order to get an SSE for security games? Not quite: in order to convert the SGM into a standard normal-form game we get a combinatorial blow-up: consider that a pure strategy would be a deployment of resources to targets. But now let's say that we just have d patrols as our resources and k targets, and a simple model where each patrol can cover exactly one target. In that case we have $\binom{k}{d}$ pure strategies for the leader. A similar blow-up happens for other natural setups such as when each resource can cover two targets (e.g. air marshals that protect an outgoing and then ingoing flight as their daily routine).

In the special case where each resource covers exactly one target (equivalently, schedules have size 1) there is an LP-based approach that can still construct an SSE in polynomial time. This LP still allows heterogeneous resources; below we let $A(r)$ be the set of targets that resource r is allowed to cover. The key idea in the LP is to use the marginal coverage probability

$p_c(t|x)$ as our decision variable. We will have an LP where the variable c_t is the coverage probability on target t, and the variable $c_{r,t}$ is the probability that resource r provides coverage for $t \in A(r)$. The goal is to maximize the defender utility subject to making some target t^* a best response for the attacker. We can then solve for each $t^* \in T$ as before, and pick the best. In this LP, we let $u_a(t|c) = c_t u_a^c(t) + (1 - c_t)u_a^u(t)$, with $u_d(t|c)$ defined analogously.

$$\begin{aligned}
\max_{c \geq 0} \quad & u_d(t^*|c) & & \\
s.t. \quad & c_t = \sum_{r \in R \text{ s.t. } t \in A(r)} c_{r,t} & \leq 1, & \quad \forall t \in T \\
& \sum_{t \in A(r)} c_{r,t} & \leq 1, & \quad \forall r \in R \\
& u_a(t|c) & \leq u_a(t^*|c), & \quad \forall t \in T.
\end{aligned} \tag{9.4}$$

This LP is polynomial in size, even though the set of pure strategies is exponential in size. It is however not immediately obvious whether the given coverage probabilities are implementable. It turns out that they are, and this can be shown via the famous Birkhoff-von Neumann theorem. Before stating the theorem, we need the definition of a *doubly substochastic matrix*, which is a matrix $M \in \mathbb{R}^{m \times n}$ such that all entries are nonnegative, each row sums to at most 1, and each column sums to at most 1.

Theorem 9.2 (Birkhoff-von Neumann theorem) *If M is doubly substochastic, then there exist matrices $M_1, M_2, \ldots, M_q$, and weights $w_1, w_2, \ldots, w_q \in (0, 1]$, such that:*

(i) $\sum_k w_k = 1$.
(ii) $\sum_k w_k M_k = M$.
(iii) *For all k, M_k is doubly substochastic, and all entries are in* $\{0, 1\}$.

Informally, the theorem states that if we have a doubly substochastic matrix, then it is possible to express it as a convex combination of "pure" or $\{0, 1\}$ doubly substochastic matrices.

The coverage probabilities $c_{r,t}$ from our LP can be viewed as a matrix with rows corresponding to resources and columns corresponding to targets. By the constraints in our LP, that matrix is doubly substochastic. It follows from the Birkhoff-von Neumann theorem that there exists pure-strategy matrices M_k (they are pure strategies by the 3rd condition of the theorem) such that their convex combination under the weight vector w adds up the correct coverage probabilities (by the 2nd condition of the theorem). By the first condition, the vector w defines a mixed strategy.

One final worry is that we don't know how large q will be in our application of the Birkhoff-von Neumann theorem. Luckily, it turns out one can show that q is $O((m+n)^2)$, and the corresponding M_k, w_k can be found in $O((m+n)^{4.5})$ time using the Dulmage-Halperin algorithm.

Unfortunately, in the more general case where schedules may cover more than one target the trick using marginal coverage probabilities turns out to fail. In that case, computing an SSE turns out to be NP-hard. Still, in practice we are usually in some variant of the hard case. There are a variety of mixed-integer programming approaches that have been used to handling this case, usually leading to acceptable performance on the real-world instances at hand. Typically, these approaches are tailored to the specific application, in order to get the most compact formulation. For that reason we will not cover them here.

9.3 Criticisms of Security Games

In security games we make a number of assumptions that can easily be critiqued: first, we assume that the attacker perfectly observes the defender strategy. Secondly, the defender knows exactly what the utility function of the attacker is (and SSE relies heavily on this). Thirdly, we assume that the attacker is perfectly rational. There are ways to address these assumptions. For example, a lot of work has gone into modeling adversaries in a way that is robust either to misspecification of the utility functions, or robust to followers not being perfectly rational.

9.4 Bayesian Games

One way to deal with uncertainty around follower utility is to assume that each player has a parameterized utility function $u_i(\cdot, \cdot|\theta_i)$, where $\theta_i \in \Theta_i$ is the *type* of player i. In Bayesian games, we assume each player draws their type from a pair of known distributions over Θ_ℓ, Θ_f. The player observes their own type before choosing an action, but not the type of the follower.

It turns out that in the special case where the follower has a single type θ_f and the leader has a probability mass $p_\ell(\theta)$ over a finite set Θ_ℓ, the LP approach for normal-form games can easily be extended to this setting and yields an optimal strategy for the leader. However, the more interesting case where the follower has multiple types is unfortunately NP-hard.

9.5 Historical Notes

The Stackelberg game model was introduced by von Stackelberg (1934) in order to analyze competing firms and first-mover advantage.

The foundations for the use of Stackelberg equilibrium in security games were laid by von Stengel and Zamir (2010)(an early version appeared online in 2004) who showed that it always helps to commit to a strategy, as long as mixed strategies are allowed, and Conitzer and Sandholm (2006) who gave efficient algorithms and complexity results around computing Stackelberg equilibrium for various game models.

In the context of security, Stackelberg equilibrium was first applied to airport security at Los Angeles International Airport Pita et al. (2008), and has since been applied to problems such as preventing poaching and illegal fishing Fang et al. (2015) and airport security screening Brown et al. (2016).

The approach based on representing strategies in terms of the marginal probability of coverage was introduced by Kiekintveld et al. (2009), and the results on polynomial-time algorithms and computational complexity in this model were given by Korzhyk et al. (2010).

Further reading.

von Stengel and Zamir (2010) is a great read for a thorough treatment of a "linear optimization" approach to understanding the mathematical structure of Stackelberg equilibria. For the use of Stackelberg equilibrium in infrastructure protection and security games, Tambe (2011) collects many of the early application papers in this area. An overview of deployed systems and new directions can be found in Sinha et al. (2018).

10
Fixed-Point Theorems and Equilibrium Existence

In this chapter we study fixed-point theorems, which are a critical tool in economics for showing the existence of a variety of equilibria. Given a function $\phi : X \to X$ mapping a space into itself, a fixed point is a point x such that $\phi(x) = x$. For a set-valued function $\phi : X \to \mathcal{P}(X)$, a fixed point is a point such that $x \in \phi(x)$. In earlier chapters, we mostly deduced the existence of equilibria in an algorithmic fashion. For example, in Chapter 4 we showed von Neumann's minimax theorem via regret minimization. In Chapter 11 we will show the existence of Fisher market equilibrium through a constructive convex program. However, in some settings we do not have algorithmic approaches for finding equilibria, yet we may wish to show that equilibria are at least guaranteed to exist, even if we do not know how to find them. The standard way to show equilibrium existence in such settings is through fixed-point theorems. We will introduce the most widely-used fixed-point theorems and show how they can be used to prove the existence of both game-theoretic equilibria and market equilibria. We will not give proofs of the fixed-point theorems themselves, which are quite technical and outside the scope of this book.

10.1 Brouwer's Fixed-Point Theorem

Brouwer's fixed-point theorem is a theorem asserting that a continuous function ϕ that maps a convex compact set unto itself is guaranteed to have a fixed point.

Theorem 10.1 *Let $X \subset \mathbb{R}^n$ be a nonempty, convex, and compact set. Let $\phi : X \to X$ be a continuous function mapping X to itself. Then there exists a point $x^* \in X$ such that $\phi(x^*) = x^*$.*

Now let us see how one can use Brouwer's fixed-point theorem to show the existence of a Nash equilibrium. Consider a normal-form game (as defined in

Chapter 2) with n players, where player i has a finite set of actions A_i and a utility function $u_i : A_1 \times \ldots \times A_n \rightarrow \mathbb{R}$. Let Δ_i be the simplex over A_i, i.e. the set of probability distributions over A_i. For a strategy profile $\sigma = (\sigma_1, \ldots, \sigma_n) \in \Delta_1 \times \ldots \times \Delta_n$, let $u_i(\sigma)$ be the expected utility of player i.

We need to construct a map $\phi : \times_{i=1}^n \Delta_i \rightarrow \times_{i=1}^n \Delta_i$ that maps a strategy profile σ to a new strategy profile σ', such that every fixed point of the map is a Nash equilibrium. Intuitively, we would like our map to be some form of *best-response mapping*, meaning that $\phi(\sigma)$ is a best response to σ for each player. Such a map would immediately satisfy the condition that fixed points are Nash equilibria. However, the best-response map is not continuous (for example, if two actions are tied as best responses, then an arbitrarily-small perturbation will break the tie), so it will not allow us to invoke Brouwer's theorem. The next section present *Kakutani's fixed-point theorem*, which enables working directly with the best response map.

In order to use Brouwer's fixed-point theorem, we will construct a "better response" map ϕ which will be continuous, while retaining the property that every fixed point of ϕ is a Nash equilibrium. We will specify ϕ on a per-player-per-action basis. For a player i and action a, let ϕ_{ia} be the following updated probability on action a:

$$\phi_{ia}(\sigma) = \frac{\sigma_{ia} + \max(0, u_i(a, \sigma_{-i}) - u_i(\sigma))}{\sum_{a' \in A_i} \sigma_{ia'} + \max(0, u_i(a', \sigma_{-i}) - u_i(\sigma))}.$$

Notice that this map increases the probability of every action a such that it is a better response to the current strategy profile σ, and decreases the probability of all other actions (through renormalization). The denominator ensures that the probabilities sum to 1.

First we convince ourselves that fixed points correspond to Nash equilibria. If σ is not a Nash equilibrium, then at least one of the increments is strictly positive, and thus $\phi(\sigma) \neq \sigma$. If $\sigma = \phi(\sigma)$, then all the increments are zero (otherwise at least one action would have its probability increased). But this implies that for each player i and every action $a \in A_i$ played with nonzero probability, it must be a best response to σ_{-i}, otherwise the increment would be positive. It follows that σ is a Nash equilibrium.

Now we want to apply Brouwer's theorem to conclude that a Nash equilibrium is guaranteed to exist. To do so, we need to show that all the preconditions are met. It is straightforward to show continuity: the max operator is continuous, the sum of a pair of functions is continuous, and the division of two continuous functions is continuous as long as the denominator is nonzero. Obviously the product set $\times_{i=1}^n \Delta_i$ is nonempty, compact and convex, and the map ϕ maps the set of strategy profiles to itself. It follows that we can apply Brouwer's fixed-point theorem to conclude that there exists a Nash equilibrium.

The idea of setting up a "better response map" is often an easy way to prove existence of equilibria in games and market models via Brouwer's theorem.

10.2 Kakutani's Fixed-Point Theorem

Kakutani's fixed-point theorem is a generalization of Brouwer's theorem, which will allow us to work directly with the best-response map. Kakutani's theorem works with set-valued mappings: the mapping ϕ maps a point x to a set $\phi(x) \subseteq X$. Set-valued mappings like this are known as *correspondences* in the economics literature. In the context of Nash equilibrium, ϕ will map a strategy profile σ to the set of best responses to σ. Because we are now working with set-valued mappings, we will need a new notion of continuity, called upper hemicontinuity. We will use a definition of upper hemicontinuity that is based on the notion of a *closed graph*. The typical definition of upper hemicontinuity is a more complicated definition based on open sets; we prefer the simpler definition based on having a closed graph (see Appendix A.5 for the usual definition of upper hemicontinuity). The closed-graph property also has the benefit of being useful in our eventual existence proof. Upper hemicontinuity and having a closed graph are equivalent properties when the correspondence is compact-valued (this equivalence is known as the *closed graph theorem*). The correspondence is indeed compact-valued in the settings that we will consider, because we will assume compactness of X.

The graph of ϕ is the set

$$\{(x, y) \in X \times X : y \in \phi(x)\}.$$

The *closed graph theorem* states that for compact-valued correspondences, upper hemicontinuity is equivalent to this graph being closed:

Theorem 10.2 *Let ϕ be a set-valued mapping from X to $\mathcal{P}(X)$, where X is compact and $\phi(x)$ is closed for all $x \in X$. Then ϕ is upper hemicontinuous if and only if its graph is closed in $X \times X$.*

Thus, we use having a closed graph as our working definition of upper hemicontinuity. Kakutani's fixed-point theorem is as follows:

Theorem 10.3 *Let $X \subset \mathbb{R}^n$ be a nonempty, convex, and compact set. Let $\phi : X \to \mathcal{P}(X)$ be a set-valued mapping such that:*

- *For every $x \in X$, $\phi(x)$ is nonempty, closed, and convex.*

- *ϕ is upper hemicontinuous at every point in X.*

Then there exists a point $x^* \in X$ such that $x^* \in \phi(x^*)$.

We can now prove the existence of a Nash equilibrium easily. We let $X = \Delta_1 \times \ldots \times \Delta_n$ be the product of the strategy spaces of the players. For a strategy profile $\sigma \in X$, let $\phi(\sigma)$ be the set of best responses to σ. Obviously X is nonempty, convex, and compact. For every $\sigma \in X$, there is always at least one best response for each player, so $\phi(\sigma)$ is nonempty. Convexity and closedness follow from the fact that the set of best responses is the convex hull of the finite set of pure best responses.

Finally, we show upper hemicontinuity using the closed graph theorem. First, note that the closed graph theorem applies, since the best response mapping is compact-valued. Thus, it suffices to show that the graph of ϕ is closed. Consider any sequence of strategy profiles converging to a point σ, and a corresponding sequence of best responses converging to a point σ'. Then it follows immediately from the continuity of expected utility that we have $\sigma' \in \phi(\sigma)$. This proves Nash's theorem.

Existence Theorem for Convex Games

Kakutani's theorem allows us to extend the above proof to showing existence of pure-strategy Nash equilibria in a class of games that we call *convex games*. Convex games are a more general class of games where each player has a convex compact strategy space, and the utility function is continuous in the strategies of the players, and quasi-concave in the player's own strategy. Convex games generalize normal-form games by treating the set of mixed-strategies in the normal-form game as the set of pure strategies in the corresponding convex game.

Theorem 10.4 *Consider a game with n players, strategy space A_i, and utility function $u_i(a_i, a_{-i})$. A pure-strategy Nash equilibrium exists if the following conditions are satisfied:*

- *A_i is convex, compact, and nonempty for all i,*
- *$u_i(s_i, \cdot)$ is continuous in s_{-i},*
- *$u_i(\cdot, s_{-i})$ is continuous and quasi-concave in s_i.*

The proof of this theorem is similar to the proof of Nash's theorem via Kakutani's fixed-point theorem. Let $X = \times_{i=1}^{n} A_i$ be the set of all strategy profiles. First note that X is nonempty, convex, and compact. For a strategy profile $s \in X$, let $\phi(s)$ be the set of best responses to s. The set $\phi(s)$ is nonempty since there is always at least one best response by continuity of utilities and the compactness of each action set. The set $\phi(s)$ is closed by continuity of $u_i(\cdot, s_{-i})$. To see that $\phi(x)$ is convex, notice that for a given player i and a set of two best responses $a, b \in \phi(\sigma)$ it must be that $u_i(a, \sigma_{-i}) = u_i(b, \sigma_{-i})$, and thus quasi-concavity implies that the same value is attained for any convex combination of a and b. For upper hemicontinuity, we can use the closed graph theorem as before, in which case the continuity of the utility functions implies that the graph of the best response mapping is closed.

10.3 Existence Theorems for Market Equilibria

In Part THREE we will study market equilibria, largely focusing on algorithmic approaches. Here, we briefly introduce a simple exchange economy and show how to prove existence of a market equilibrium using the theory developed in this chapter. In market equilibrium problems, we have a set of buyers, and a set of items. The goal is to find a set of *prices* for the items, such that the market clears: when we allow each buyer to purchase their favorite bundle under the stated prices, the total demand for each item equals the total supply of that item. In the Fisher market model that we will study later, each buyer is endowed with a budget, which restricts the set of feasible allocations that a buyer can purchase given the prices. In this section, we consider a more general *exchange model* where each buyer is endowed with an initial bundle of items, which they get to sell for the prices that are set by the market, and their income from selling in turn defines their budget constraint for purchasing other items in the market. The Fisher market model is a special case of this model where each buyer is endowed with an equal amount of every item proportional to their budget.

Suppose we have n buyers and m items. For a given bundle of items $x \in \mathbb{R}^m_{\geq 0}$, buyer i has a utility function $u_i(x)$ that is strictly concave, continuous, and strictly monotonic (i.e. $u_i(x) > u_i(x')$ if $x \geq x'$ and $x_j > x'_j$ for some j). Each buyer is endowed with an amount $\omega_i \in \mathbb{R}^m_{\geq 0}$ of each item. Given a set of prices $p \in \mathbb{R}^m_{\geq 0}$, we can define the *demand* of buyer i as the solution to the following optimization problem (which has a unique solution due to strict concavity):

$$D_i(p) = \arg \max_{x'_i \in \mathbb{R}^m_{\geq 0}} u_i(x_i) \text{ s.t. } \langle p, x'_i \rangle \leq \langle p, \omega_i \rangle. \tag{10.1}$$

In market equilibrium models it is useful to define and work with the *aggregate demand* function $z(p) = \sum_{i=1}^n D_i(p)$, which is the total demand for each item in the market at a given price vector p. A market equilibrium is then a set of prices p such that the market clears, i.e. $0 \in z(p)$. In general the aggregate demand function could be set-valued, since the demand of each buyer is potentially set-valued. Because we assumed strict concavity of each buyer's utility function the aggregate demand function output is a single point, since the demand of each buyer is unique for any price vector p. If the demands are not unique then additional smoothing tricks are required in order to prove equilibrium existence.

First we state an abstract existence theorem for single-valued aggregate demand functions. Then we will show how to apply it to our exchange economy. Consider a function $z : \mathbb{R}^m_+ \to \mathbb{R}^m \cup \{+\infty\}^m$ that maps a price vector p to an aggregate demand vector $z(p)$.

Definition 10.5 We say that the aggregate demand function z is *well-behaved* if it satisfies the following properties:

(i) Continuity: $z(p)$ is continuous in p.
(ii) Homogeneity: $z(p)$ is zero'th order homogeneous, i.e. $z(\alpha p) = z(p)$ for any scalar $\alpha > 0$.
(iii) Walras' law holds: $\langle p, z(p) \rangle = 0$.
(iv) Lower bounded: there exists a constant $s > 0$ such that for all $p \in \mathbb{R}^n_{\geq 0}$, $z(p) \geq -s$.
(v) Unbounded demand: If $\{p^t\}_{t=1}^{\infty}$ is a sequence of price vectors converging to $p \neq 0$, with $p_j = 0$ for some j, then $\|z(p^t)\|_1 \to \infty$.

Theorem 10.6 *Any well-behaved demand function z has a price vector $p^* \in \mathbb{R}^n_{\geq 0}$ such that $z(p^*) = 0$.*

We will not give the full proof of this theorem, but we will sketch the main ideas. The reader is encouraged to finish the proof on their own. The first thing to notice is that, from homogeneity, we can restrict the price vector to lie in the unit simplex $\Delta^m = \{p \in \mathbb{R}^m_{\geq 0} : \|p\|_1 = 1\}$ (if $z(p) = 0$ then $z(p/\|p\|_1) = 0$). Then we define a map $\phi : \Delta^m \to \Delta^m$ as follows:

$$\phi(p) = \begin{cases} \arg\max_{q \in \Delta^m} \langle q, z(p) \rangle & \text{if } p > 0, \\ \{q \in \Delta^m : \langle q, p \rangle = 0\} & \text{if } p_j = 0 \text{ for some } j. \end{cases}$$

For a set of items in the interior of the simplex, this price mapping is a "best-response" mapping if we imagine a seller of the items that tries to maximize revenue given the stated demand. For a price vector p where $p_j = 0$ for some j, we will have infinite demand due to condition (v) of Definition 10.5, and therefore many price vectors would achieve infinite revenue. The definition of ϕ restricts the output to be a price vector that only puts positive price on items with infinite demand (a subset of the vectors that achieve infinite revenue). With this setup, one can apply Kakutani's fixed-point theorem.

Next, we show how to apply this theorem to our exchange economy by showing that our demand function is well-behaved. Since the utility function is strictly concave, we have that the demand in Eq. (10.1) for each buyer is unique for any price vector p. Therefore, the aggregate demand function is well-defined and has a unique output. Continuity of the demand function follows from *Berge's maximum theorem* (see Theorem A.8), which is a theorem guaranteeing continuity properties of parameterized optimization problems. Berge's maximum theorem is a very useful tool for analyzing problems in economics. Specifically, Berge's maximum theorem for compact convex programs with a strictly convex objective guarantees that the optimal solution is a continuous function of the input parameters. In our case the strictly convex program in question is the demand problem for each buyer, whose feasible set is parameterized by the price vector p.

The demand function is also zero'th order homogeneous: if we scale the price vector by a constant then we scale both the left and right-hand side of the constraint by the same constant, and thus the feasible set is unchanged.

Walras' law holds because the budget of each buyer is equal to the total value of their endowment and their utility is strictly increasing in consumption, so they must spend their whole budget. From these observations, we have $\langle p, \sum_{i=1}^n D_i(p)\rangle = \langle p, \sum_{i=1}^n \omega_i\rangle$. Subtracting the two equalities gives Walras' law.

The aggregate demand function is lower bounded because demands are nonnegative, and thus the aggregate demand is bounded below by the sum of the endowments.

For unbounded demand, consider a sequence of price vectors p^t converging to $p \neq 0$, with $p_j = 0$ for at least one j. Then there is at least one buyer i whose budget is bounded below by a strictly positive constant for all $t \geq t_0$, for some large enough t_0. Now suppose for contradiction that this buyer's demand is bounded above by a constant for all t. In that case, there must be a convergent subsequence of demands for that buyer. Let $\{x_i^\tau\}$ be the converging subsequence of demands for buyer i and let $\bar{x}_i$ be the limit point. Now consider the utility $u_i(\bar{x}_i)$ that buyer i gets from the limit point. Suppose we give buyer i an additional unit of item j, then this new allocation $\bar{x}_i + e_j$ is still budget feasible under the limit price p, and we have increased the utility of buyer i by some $\epsilon > 0$. By lower hemicontinuity of the set of budget-feasible allocations, there must exist a sequence of allocations $\hat{x}_i^\tau$ converging to the allocation $\bar{x}_i + e_j$ such that each $\hat{x}_i^\tau$ is budget feasible under the price vector p^{t_τ}. But then by the continuity of the utility function we have a contradiction, since this implies that x_i^τ is not utility maximizing for some sufficiently large τ.

10.4 Historical Notes

Brouwer's fixed-point theorem was originally proven by Dutch mathematician and philosopher L.E.J. Brouwer for the special case where X is a unit ball. The generalization follows by using homomorphism to map the unit ball to any convex compact set. Nash's theorem, which guarantees the existence of an equilibrium in a finite game, was originally proved using Kakutani's fixed-point theorem (Nash Jr, 1950), which it itself a generalization of Brouwer's fixed-point theorem. Interestingly, John von Neumann had proven a generalization of Brouwer's fixed-point theorem in 1937 (Neumann, 1937), but it was much less straightforward to apply, and Kakutani's fixed-point theorem is a simplified and easier-to-apply version of von Neumann's result. One of the most foundational results in economics, the existence theorem for a competitive equilibrium in the Arrow-Debreu model of a competitive economy (Arrow and Debreu, 1954) was proven using an equilibrium existence theorem developed by Debreu (1952). This theorem, in turn, utilized a generalization of Kakutani's fixed-point theorem to non-convex sets (Begle, 1950; Eilenberg and Montgomery, 1946).

Further reading

Ok (2011) is a good starting point for a more in-depth study of fixed-point theorems in finite-dimensional settings (and real analysis for economics more broadly). For a very comprehensive treatment of fixed-point theorems and their economic use cases, see Aliprantis and Border (2006). That book is particularly useful in the context of infinite-dimensional games and economies.

PART THREE

FAIR ALLOCATION AND MARKET EQUILIBRIUM

11
Fair Division and Market Equilibrium

In this chapter we start the study of fair allocation of resources to a set of individuals. We start by focusing on the *fair division* setting. In fair division, we have one or more items that we wish to allocate to a set of agents, under the assumption that the items are infinitely-divisible, meaning that we can perform fractional allocation. In the next chapter we will study the setting with discrete items. The goal will be to allocate the items in a manner that is efficient, while attempting to satisfy various notions of fairness towards each individual agent. Fair allocation has many applications such as assigning course seats to students, pilot-to-plane assignment for airlines, dividing estates, dividing chores among a household, rent division for roommates, and fair recommender systems.

We study fair division problems with the following setup: we have a set of m infinitely-divisible items that we wish to divide among n agents. Without loss of generality we may assume that each item has supply 1. We shall use $x \in \mathbb{R}_{\geq 0}^{n \times m}$ to denote an assignment of items to agents, where x_{ij} is how much agent i gets of item j We will denote the bundle of items given to agent i as $x_i \in \mathbb{R}_{\geq 0}^{m}$. Each agent has some utility function $u_i(x_i) \in \mathbb{R}_{\geq 0}$ denoting how much they like the bundle x_i.

Given the above, we would like to choose a "good" assignment x of items to agents. However, "good" turns out to be very complicated in the setting of fair division, as there are many possible desiderata we may wish to account for.

First, we would like the allocation to somehow be efficient, meaning that it should lead to high utilities for the agents in aggregate. One option would be to try to maximize the *social welfare* $\sum_i u_i(x_i)$, the sum of agent utilities. However, this turns out to be incompatible with the fairness notions that we will introduce later. An easy criticism of social welfare in the context of fair division is that it favors *utility monsters*: agents with much greater capacity for utility are given more items[1]. Instead, we shall focus on the weaker notion of *Pareto optimality*: we wish to find an allocation x such that for *every* other allocation x', if some agent i' is better off under x', then some other agent is strictly worse off. In other words, x should be such that no other allocation x' is "strictly better," where strictly better means that $u_i(x') \geq u_i(x)$ for all $i \in [n]$, and $u_i(x') > u_i(x)$ for some $i \in [n]$.

We will consider the following measures of how fair an allocation x is:

- *No envy*: x has no envy if for every pair of agents i, i', $u_i(x_i) \geq u_i(x_{i'})$. In other words, every agent likes their own bundle at least as much as that of anyone else.
- *Proportionality*: x satisfies proportionality if $u_i(x_i) \geq u_i\left(\vec{1} \cdot \frac{1}{n}\right)$. That is, every agent likes their bundle x_i at least as well as the bundle where they receive $\frac{1}{n}$ of every item. This property is also sometimes known as the *fair shares* property. The fair share name evokes the following reasoning for insisting on proportionality: absent any other information or valuations, the most natural way to divide items would be to simply say that each agent is entitled to an equal share of each item. Proportionality ensures that agents are at least as happy as under such an equal shares allocation.

In the case of $n = 2$, no envy and proportionality are equivalent. More generally, no envy is a stronger guarantee than proportionality, in the sense that no envy implies that proportionality is satisfied. It is a useful exercise to come up with an argument for these relationships.

We begin our study of fair division mechanisms with a classic approach: *competitive equilibrium from equal incomes* (CEEI). In CEEI, we construct a mechanism for fair division by giving each agent a unit budget of fake currency (or *funny money*), computing what is called a competitive equilibrium (also known as *Walrasian equilibrium* or *market equilibrium*; we will use the term market equilibrium) under this new market, and using the corresponding allocation as our fair division. The fake currency is then thrown away, since it had no purpose except to define a market.

To understand this mechanism, we first introduce *market equilibrium*. We focus on a *Fisher market* economy, where there is a set of buyers $[n]$ (we use *buyer* and *agent* interchangeably), each with some budget B_i, and a set of items $[m]$ being sold (we assume that the supply of each item is one without loss of generality). In a market equilibrium, we wish to find a set of prices $p \in \mathbb{R}^m_{\geq 0}$ for each of the m items such that the market *clears*. Intuitively, a market clears when there exists an allocation x of items to buyers such that everybody is

[1] See also `https://existentialcomics.com/comic/8`

assigned an optimal allocation given the prices and their budget, and all items are exactly allocated at their supply. Formally, the *demand set* of a buyer i with budget B_i is

$$D_i(p) = \arg\max_{x_i \geq 0} u_i(x_i) \text{ s.t. } \langle p, x_i \rangle \leq B_i.$$

Notice that the demand set is only indirectly dependent on the utility function u_i. In particular, we will mostly focus on utilities that are homogeneous of degree one, in which case the demand function D_i is invariant to scaling the buyer's utility function by a positive constant.

Definition 11.1 A market equilibrium is an allocation $x \in \mathbb{R}^{n \times m}_{\geq 0}$ and a price vector $p \in \mathbb{R}^m_{\geq 0}$ such that:

(i) Demands are satisfied: $x_i \in D(p)$ for all buyers i.
(ii) The market clears: $\sum_{i \in [n]} x_{ij} \leq 1$, and $\sum_{i \in [n]} x_{ij} = 1$ if $p_j > 0$.

Every market equilibrium is Pareto optimal by the *first fundamental theorem of welfare economics*, under the mild condition that each utility function is *nonsatiating*. A utility function $u_i : \mathbb{R}^m_{\geq 0} \to \mathbb{R}_{\geq 0}$ is nonsatiating if for any $x_i \in \mathbb{R}^m_{\geq 0}$ and vector of additional items $\vec{\epsilon} = \epsilon \cdot \vec{1}$ where $\epsilon > 0$, we have $u_i(x + \vec{\epsilon}) > u_i(x)$. In words, buyers always strictly improve their utility when they get an arbitrarily-small additional amount of every good. To prove this, we will use an extremely useful property of market equilibria under nonsatiating utilities, which is that the sum of prices must equal the sum of the budgets:

Lemma 11.2 *Let each utility function be nonsatiating. Then for any market equilibrium* (x, p)*, we have* $\sum_{j \in [m]} p_j = \sum_{i \in [n]} B_i$*.*

Proof Every buyer always spends their budget fully, i.e. $p^\top x_i = B_i$. If not, then they could buy some additional small amount $\vec{\epsilon}$ of items and strictly improve their utility. It follows that

$$\sum_{i \in [n]} B_i = \sum_{i \in [n]} p^\top x_i = \sum_{j \in [m]} p_j \sum_{i \in [n]} x_i.$$

Now notice that $p_j \sum_{i \in [n]} x_i = p_j$, because either $\sum_{i \in [n]} x_i = 1$, or $p_j = 0$ by the market clearing condition of a market equilibrium. The lemma follows. □

Theorem 11.3 (First fundamental theorem of welfare economics) *Let each utility function be nonsatiating. Then for any market equilibrium* (x, p)*, the allocation* x *is Pareto optimal.*

Proof Let (x, p) be a market equilibrium. Now suppose that x is not Pareto optimal. Then there exists some x' such that $u_i(x'_i) \geq u_i(x_i)$ for all i, and some k such that $u_k(x'_k) > u_k(x_k)$. Now notice that $p^\top x'_i \geq B_i$ for each buyer i. If this does not hold, then buyer i can afford $x'_i + \vec{\epsilon}$ for some sufficiently-small $\vec{\epsilon} \in \mathbb{R}^m_{\geq 0}$ and strictly improve their utility, which contradicts x_i being in the demand set under p. Now consider buyer k. For buyer k, we must have $p^\top x'_k > B_k$, since otherwise they could afford x'_k under p and strictly improve their utility, again contradicting demand satisfaction.

Summing over expenditures under x', we have $\sum_{i \in [n]} p^\top x'_i > \sum_{i \in [n]} B_i$. Since x' is feasible, we have $\sum_{i \in [n]} p^\top x'_i \leq \sum_{j \in [m]} p_j$. But this implies $\sum_{j \in [m]} p_j > \sum_{i \in [n]} B_i$, which contradicts Lemma 11.2. □

Market equilibria are interesting in their own right, though we will mainly study them here in the context of the CEEI mechanism for fair allocation. CEEI is a perfect solution to the desiderata. It is Pareto optimal by Theorem 11.3. It has no envy: since each agent has the same budget $B_i = 1$ in CEEI and every agent is buying something in their demand set, no envy must be satisfied, since they can afford the bundle of any other agent. Finally, proportionality is satisfied, since each agent can afford the bundle where they get $\frac{1}{n}$ of each item (convince yourself why).

Market-equilibrium-based allocation for divisible items has applications in large-scale Internet markets. First, it can be applied in *fair recommender systems*. As an example, consider a job recommendations site. It's a two-sided market. On one side are the users, whom view job ads. On the other side are the companies creating job ads. Naively, a system might try to simply maximize the number of job ads that users click on, or apply to. This can lead to extremely imbalanced allocations, where a few job ads get a huge number of views and applicants, which is bad both for users and the companies. Instead, the system may wish to fairly distribute user views across the many job ads. In that case, CEEI can be used. In this setting the agents are the job ads, and the items are slots in the ranked list of job ads shown to the user. Secondly, there are strong connections between market equilibrium and the allocation of ads in large-scale Internet ad markets. This connection will be explored in detail in Chapter 16.

11.1 Fisher Market

We study market equilibrium in the *Fisher market* setting. As in the fair division setting, we have a set of m infinitely-divisible items that we wish to divide among n buyers, and the setup is the same with respect to supplies and utilities.

Distinctly from the fair division setting, each buyer is also endowed with a budget B_i of currency (possibly fake currency as in CEEI).

We assume throughout the chapter that there always exists a feasible allocation x such that $u_i(x_i) > 0$ for all $i \in [n]$. This is benign assumption. For example, if each buyer has a linear utility function then it merely means that for every buyer $i \in [n]$, they have strictly positive value for at least one item. If some buyer has valuation zero for *every* item then we can preprocess away that buyer, since the allocation does not matter to them.

11.1.1 Linear Utilities

We start by studying the simplest setting, where the utility of each buyer is linear. This means that every buyer i has some valuation vector $v_i \in \mathbb{R}^m$, and their utility for an allocation $x_i \in \mathbb{R}^m_{\geq 0}$ is $u_i(x_i) = \langle v_i, x_i \rangle$.

Amazingly, there is a nice convex program for computing a market equilibrium. Before giving the convex program, let us consider some properties that we would like, which will motivate the structure of the convex program. First, if we are going to find a feasible allocation, we want the supply constraint to be respected for each item $j \in [m]$, i.e.

$$\sum_{i \in [n]} x_{ij} \leq 1, \forall j.$$

Secondly, since a buyer's demand does not change even if we rescale their valuation by a constant, we would like the optimal solution to our convex program to also remain unchanged. Similarly, splitting the budget of a buyer into two separate buyers with the same valuation function should leave the allocation unchanged after allocating the items proportionally among the two split buyers. These conditions are satisfied by the budget-weighted geometric mean of the utilities:

$$\left(\prod_{i \in [n]} u_i(x_i)^{B_i} \right)^{1/\sum_{i \in [n]} B_i}.$$

Taking the objective to the $\sum_{i \in [n]} B_i$'th power does not change the set of optimal solutions (after taking this power, the resulting objective is known as *Nash welfare*). Secondly, taking the logarithm does not change the set of optimal solutions. Because we are taking the logarithm, we now have an implicit domain constraint $u_i(x_i) > 0$ for each buyer $i \in [n]$. Because we assumed that every buyer has at least one allocation with strictly positive utility, we always have at least one feasible solution.

Based on the above, maximizing the budget-weighted geometric mean is equivalent to the following convex program, known as the *Eisenberg-Gale* (EG) convex program:

$$\begin{array}{llll} \max\limits_{x \geq 0} & \sum\limits_{i \in [n]} B_i \log \langle v_i, x_i \rangle & & \text{Dual variables} \\ s.t. & \sum\limits_{i \in [n]} x_{ij} \leq 1, \quad \forall j = [m], & & \Big| \; p_j \end{array} \tag{EG}$$

On the right are the dual variables associated to each constraint. It is easy to see that this is a convex program. First, the feasible set is defined by linear inequalities. Second, we are taking a max of a sum of concave functions composed with linear maps. Since taking a sum and composing with a linear map both preserve concavity we get that the objective is concave.

The solution to the primal problem x along with the vector of dual variables p yields a market equilibrium. Here we assume that for every item j there exists i such that $v_{ij} > 0$, and every buyer values at least one item above 0.

Theorem 11.4 *The pair of allocations x and dual variables p from EG forms a market equilibrium.*

Proof To see this, we need to look at the KKT conditions (see Theorem A.4) of the primal and dual variables. We omit some KKT conditions that are not needed. The KKT conditions we use are:

(i) Complementary slackness for item j: $p_j > 0 \Rightarrow \sum_{i \in [n]} x_{ij} = 1$.
(ii) First-order optimality on x_{ij}: $\frac{B_i}{\langle v_i, x_i \rangle} \leq \frac{p_j}{v_{ij}}$.
(iii) First-order optimality when $x_{ij} > 0$: $\frac{B_i}{\langle v_i, x_i \rangle} = \frac{p_j}{v_{ij}}$.

The first KKT condition shows that every item is fully allocated, since for every j there is some buyer i with non-zero value and by the second condition $p_j \geq \frac{v_{ij} B_i}{\langle v_i, x_i \rangle} > 0$. Thus, we satisfy the market clearing condition in the definition of market equilibrium (Definition 11.1).

The other condition in Definition 11.1 is that every buyer is assigned a bundle from their demand set. We will use $\beta_i = \frac{B_i}{\langle v_i, x_i \rangle} = \frac{B_i}{u_i(x_i)}$ to denote the *utility price* that buyer i pays. First off, by the second KKT condition we have that the utility price that buyer i gets satisfies

$$\beta_i \leq \frac{p_j}{v_{ij}}.$$

By the third KKT condition, we have that if $x_{ij} > 0$ then for all other items j' we have

$$\frac{p_j}{v_{ij}} = \beta_i \leq \frac{p_{j'}}{v_{ij'}}.$$

Thus, any item j that buyer i is assigned has at least as low of a utility price as any other item j'. In other words, they only buy items that have the best bang-per-buck among all the items. It remains to show that they spend their whole budget. Multiplying the third KKT condition by x_{ij} and rearranging gives

$$x_{ij} v_{ij} \frac{B_i}{\langle v_i, x_i \rangle} = p_j x_{ij},$$

for any j such that $x_{ij} > 0$. If $x_{ij} = 0$ then $p_j x_{ij} = 0$. Summing across all $j \in [m]$ yields

$$\sum_{j \in [m]} p_j x_{ij} = \sum_{j \in [m]} x_{ij} v_{ij} \frac{B_i}{\langle v_i, x_i \rangle} = \langle v_i, x_i \rangle \frac{B_i}{\langle v_i, x_i \rangle} = B_i.$$

□

EG gives us an immediate proof of existence for the linear Fisher market setting: the feasible set is clearly non-empty, and the max is guaranteed to be achieved.

Theorem 11.3 showed that all market equilibria are Pareto optimal. It is now trivial to see that Pareto optimality holds in Fisher-market equilibrium: since it is a solution to EG, it must be. Otherwise, we could construct a solution with strictly better objective by using the allocation that yields weakly greater utility for every buyer and strictly better utility for some buyer.

From the EG formulation we can also see that the equilibrium utilities $u_i(x^*)$ and prices p^* are unique. First note that any market equilibrium allocation would satisfy the optimality conditions of EG, and thus be an optimal solution. But if there were more than one set of utility vectors that were equilibria, then by the strict concavity of the log we would get that there is a strictly better solution, which is a contradiction. That equilibrium prices are unique now follows from the third KKT condition, since all terms except the utilities are constants. The equilibrium allocation x^* is not unique, as some buyers may be indifferent between some items.

11.2 More General Utilities

It turns out that EG can be applied to a broader class of utilities. This class is the set of utilities that are continuous, concave, nonnegative, and homogeneous degree one (i.e. $u_i(\alpha x) = \alpha u_i(x)$ for $\alpha \geq 0$) (abbreviated CCNH).

In that case we get an optimization problem of the form

$$\begin{array}{llll} \max_{x \geq 0} & \sum_{i \in [n]} B_i \log u_i(x_i) & & \text{Dual variables} \\ s.t. & \sum_{i \in [n]} x_{ij} \leq 1, \quad \forall j \in [m], & & \Big| \; p_j. \end{array} \tag{EG}$$

This is still a convex optimization problem, since composing a concave and nondecreasing function (the log function) with a concave function (u_i) yields a concave function. Beyond linear utilities, the most famous classes of CCNH utilities are:

(i) Cobb-Douglas utilities: $u_i(x_i) = \prod_j (x_{ij})^{a_{ij}}$, where $\sum_{j \in [m]} a_{ij} = 1, a_{ij} \geq 0$.
(ii) Leontief utilities: $u_i(x_i) = \min_j \frac{x_{ij}}{a_{ij}}$.
(iii) The family of constant elasticity of substitution (CES) utilities: $u_i(x_i) = \left(\sum_{j \in [m]} a_{ij} x_{ij}^{\rho}\right)^{1/\rho}$, where a_{ij} are the utility parameters of a buyer, and ρ parameterizes the family, with $-\infty < \rho \leq 1$ and $\rho \neq 0$.

CES utilities turn out to generalize all the other utilities we have seen so far: Leontief utilities are obtained as ρ approaches $-\infty$, Cobb-Douglas utilities as ρ approaches 0, and linear utilities when $\rho = 1$. More generally, $\rho < 0$ means that items are complements, whereas $\rho > 0$ means that items are substitutes.

If u_i is continuously differentiable then the proof that EG computes a market equilibrium in this more general setting essentially follows that of the linear case. The only non-trivial change is that when we derive KKT conditions with respect to x_i we get

(i) $\frac{B_i}{u_i(x_i)} \leq \frac{p_j}{\partial u_i(x_i)/\partial x_{ij}}$.
(ii) $x_{ij} > 0 \Rightarrow \frac{B_i}{u_i(x_i)} = \frac{p_j}{\partial u_i(x_i)/\partial x_{ij}}$.

In order to prove that buyers spend their budget exactly in this setting we can apply Euler's homogeneous function theorem $u_i(x_i) = \sum_{j \in [m]} x_{ij} \frac{\partial u_i(x_i)}{\partial x_{ij}}$ to get

$$\sum_{j \in [m]} x_{ij} p_j = \sum_{j \in [m]} x_{ij} \frac{\partial u_i(x_i)}{\partial x_{ij}} \frac{B_i}{u_i(x_i)} = B_i.$$

From the KKT conditions and the above equality, one can conclude that the KKT conditions of the buyer's demand optimization problem are satisfied.

11.3 Computing Market Equilibrium

Now we know how to write a market equilibrium problem as a convex program. How should we solve it? One option is to build the EG convex program explicitly using mathematical programming software. A lot of contemporary software is not very good at handling this kind of objective function (formally this falls under *exponential cone programming*, which is still relatively new). In particular, the default solvers e.g. in CVXPY fail due to numerical issues for relatively small instances with around 150 items and 150 buyers. The Mosek solver is currently the only industry-grade solver that supports exponential cone programming. It fares much better, and scales to a few thousand buyers and items. For problems of moderate-to-large size, this is the most effective approach. The open-source Clarabel solver also performs quite well on solving the Eisenberg-Gale convex program, and is a good option for those who do not have access to Mosek. However, for very large instances, the iterations of the interior-point solver used in any of these solvers become too slow.

Instead, for large problems we may invoke some of our earlier results on saddle-point problems. In particular, the EG convex program is amenable to online mirror descent and the folk-theorem based approach for solving saddle-point problems (if we construct the Lagrangian based on the prices). In that framework, we can interpret the repeated game as being played between a pricer trying to minimize over the prices p, and the set of buyers choosing allocations x.

The next chapter will show how to apply OMD to the EG program. In the case of linear utilities, this leads to a particularly compelling algorithm that achieves even stronger guarantees than OMD achieved in the case of zero-sum games in Chapter 6.

11.4 Historical Notes

The original Eisenberg-Gale convex program was given for linear utilities by Eisenberg and Gale (1959). Eisenberg (1961) later extended it to utilities that are concave, continuous, and homogeneous.

Fairly assigning course seats to students via market equilibrium was studied by Budish (2011). Goldman and Procaccia (2015) introduce an online service `spliddit.org` which has a user-friendly interface for fairly dividing many things such as estates, rent, fares, and others. The motivating example of fair recommender systems, where we fairly divide impressions among content creators via CEEI was suggested in Kroer et al. (2019) and Kroer and Peysakhovich

(2019). Similar models, but where money has real value, were considered for ad auctions with budget constraints by several authors Borgs et al. (2007); Conitzer et al. (2018, 2019)

A fairly comprehensive recent overview of fair division can be found at `https://www.cs.toronto.edu/~nisarg/papers/Fair-Division-Tutorial.pdf`.

Further reading.

Nisan et al. (2007) has two good chapters on the Eisenberg-Gale convex program and market equilibrium computation. For market equilibrium, economics textbooks typically focus on more general cases than the Fisher market, such as the Arrow-Debreu model of general equilibrium. A good reference for this is Mas-Colell et al. (1995), which has a very comprehensive treatment of general equilibrium theory, including existence and uniqueness of equilibria, welfare theorems, and applications. For fair division, the book Brams and Taylor (1996) has in-depth coverage from an economic perspective.

12
Computing Fisher Market Equilibrium

In this chapter we will look at methods for computing market equilibrium at scale. In particular, we will consider two complementary approaches: 1) how to run fast iterative methods in order to compute a market equilibrium, and 2) how to abstract the market, either down to a manageable size, or in order to deal with incomplete valuations. The setup will be the same as in Chapter 11.

12.1 Interlude on Convex Conjugates

Given a function $f : \mathbb{R}^n \to \mathbb{R}$ we say that its *convex conjugate* is the function

$$f^*(y) = \sup_x \langle y, x \rangle - f(x).$$

We will be interested in the convex conjugate of the function $f(x) = -\log x$. We get

$$f^*(y) = \sup_x \; yx + \log x.$$

For $y \geq 0$, we get that $f^*(y) = +\infty$. For $y < 0$, using first-order optimality we get $x^* = -1/y$. It follows that

$$f^*(y) = -1 + \log(-1/y) = -1 - \log(-y). \tag{12.1}$$

12.2 Duals of the Eisenberg-Gale Convex Program

In Chapter 11 we saw that the Eisenberg-Gale convex program (EG) yields a market equilibrium for Fisher markets with linear utilities. Recall that x_i is the allocation for buyer i. Overloading notation slightly, we introduce a variable u_i

which will represent the utility $u_i(x_i)$. We also add a constraint which enforces this interpretation. This yields the following program:

$$
\begin{array}{lllll|l}
\max\limits_{x\geq 0} & \sum\limits_{i\in[n]} B_i \log u_i & & & & \text{Dual variables} \\
s.t. & u_i & \leq \langle v_i, x_i\rangle, & \forall i = 1,\dots,n, & & \beta_i \\
& \sum\limits_{i\in[n]} x_{ij} & \leq 1, & \forall j = 1,\dots,m, & & p_j.
\end{array}
$$

The u_i variable and its corresponding constraint are redundant, but this rewrite is very useful for deriving the dual of the EG program (even in the case of nonlinear utilities this trick is almost always the easiest way to derive the dual).

We will now derive the dual, and eventually use a further duality step to derive an interesting and very practical algorithm for solving EG. We introduce dual variables β_i (corresponding to the utility price of buyer i), and p_j (the price of item j). The dual variables are listed on the right of their corresponding primal constraint. We construct the Lagrangian

$$L(x,u,\beta,p) = \sum_{i\in[n]} B_i \log u_i + \sum_{i\in[n]} \beta_i(\langle v_i, x_i\rangle - u_i) + \sum_{j\in[m]} p_j(1 - \sum_{i\in[n]} x_{ij}).$$

The standard Lagrangian dual is then

$$\min_{p\geq 0,\beta\geq 0} \max_{x\geq 0,u} L(x,u,\beta,p) \tag{12.2}$$

Now, we simplify the inner max and use the notation $\delta[\cdot]$ to denote an indicator function of whether a given condition is true:

$$
\begin{aligned}
\max_{x\geq 0,u} L(x,u,\beta,p) &= \sum_{j\in[m]} p_j + \sum_i \left[\max_{u_i} (B_i \log u_i - \beta_i u_i) + \max_{x_i\geq 0} \langle \beta_i v_i - p, x_i\rangle\right] \\
&= \sum_{j\in[m]} p_j + \sum_i \left[\max_{u_i} (B_i \log u_i - \beta_i u_i) + \delta\left[\beta_i v_i \leq p\right]\right] \\
&= \sum_{j\in[m]} p_j + \sum_i \left[B_i \max_{u_i} \left(\log u_i - \frac{\beta_i}{B_i} u_i\right) + \delta\left[\beta_i v_i \leq p\right]\right] \\
&= \sum_{j\in[m]} p_j + \sum_i \left[B_i\left(-1 - \log \beta_i + \log B_i\right) + \delta\left[\beta_i v_i \leq p\right]\right].
\end{aligned}
$$

The first equality is by rearranging terms. The second equality is by noting that the max over $x_i \geq 0$ is positive infinity if $\beta_i v_{ij} > p_j$ for any j. The third equality is by rearranging B_i. The fourth equality is by (12.1).

Thus, we get that the dual (12.2) is equal to

$$\begin{aligned} \min_{p\geq 0, \beta \geq 0} \quad & \sum_j p_j - \sum_{i\in[n]} B_i \log(\beta_i) + \sum_{i\in[n]} (\log B_i - B_i) \\ & p_j \geq v_{ij}\beta_i, \quad \forall i, j. \end{aligned} \tag{12.3}$$

Finally, we may drop the terms $\sum_{i\in[n]} (\log B_i - B_i)$ since they are constant. This yields the standard dual of EG (whose objective at optimality is equal to the primal objective at optimality, up to the addition of the constants):

$$\begin{aligned} \min_{p\geq 0, \beta \geq 0} \quad & \sum_{j\in[m]} p_j - \sum_{i\in[n]} B_i \log(\beta_i) \\ & p_j \geq v_{ij}\beta_i, \quad \forall i, j. \end{aligned} \tag{12.4}$$

The EG dual can be used to derive several attractive algorithms for computing Fisher market equilibria. Many of these algorithms are based on the fact that we can reformulate away either the β variables or the p variables in Eq. (12.4). To reformulate away the β variables, notice that for each $i \in [n]$, we want to make β_i as large as possible. For a fixed set of prices p, this is achieved by setting $\beta_i = p_j/v_{ij}$. Applying the reformulation yields the following dual program, which is only concerned with the prices:

$$\min_{p\geq 0} \quad \sum_j p_j - \sum_{i\in[n]} B_i \log(\min_{j\in[m]} p_j/v_{ij}). \tag{12.5}$$

If we apply OGD to the prices of this program, we get the following price update dynamics:

$$p^{t+1} = p^t - \eta\left(\sum_{i\in[n]} D_i(p^t) - 1\right),$$

where $D_i(p^t)$ should be interpreted as selecting an arbitrary element of the demand set for each buyer i. This algorithm is an example of a classic idea from economics known as tâtonnement. In tâtonnement, the idea is that the market is adaptively updating the prices based on observing the aggregate demand of the buyers, and pushing prices in the opposite direction (e.g. if an item is overdemanded, then the market increases prices, and vice versa). If natural price-adjustment dynamics such as tâtonnement converge, then it can be seen as a justification for how a market might arrive at equilibrium prices without central coordination. We have shown that for Fisher markets, a very simple tâtonnement update arises directly by applying OGD (equivalently, subgradient descent) to the EG dual.

To instead reformulate away the prices in Eq. (12.4), we can use the fact that for a fixed β vector, each price p_j should be made as small as possible, which is achieved by setting $p_j = \max_{i\in[n]} \beta_i v_{ij}$. This yields the following program:

$$\min_{\beta\geq 0} \quad \sum_{j\in[m]} \max_{i\in[n]} \beta_i v_{ij} - \sum_{i\in[n]} B_i \log(\beta_i) \tag{12.6}$$

Eq. (12.6) can be used to derive a very natural auction-based dynamics. Given some current β^t, for each item j we get that the subdifferential of the first term in Eq. (12.6) is the set of vectors x^t_j such that $\sum_{i\in[n]} x^t_{ij} = 1$, and $x^t_{ij} > 0$ only when $i \in \arg\max_{k\in[n]} \beta^t_k v_{ik}$. Thus, we have that the item is allocated to buyers with the highest "bid" $b_{ij} = \max_{i\in[n]} \beta^t_i v_{ij}$, and the price is set equal to the highest bid. This is exactly the first-price auction rule from Chapter 3. This leads to what is known as the PACE dynamics, where each buyer updates their β^{t+1}_i using the formula $\beta^{t+1}_i = B_i/\bar{u}^t_i$, where $\bar{u}^t_i = (1/t)\sum_{t\in[T]}\langle v_i, x^t_i\rangle$ is the average utility the buyer has received across time. PACE arises by applying the follow-the-leader (FTL) update (see Chapter 4) to the sequence of utilities observed by each buyer. While FTL did not work in the general no-regret learning environment of Chapter 4, the curvature induced by the $B_i \log(\beta_i)$ term is enough to recover convergence.

12.2.1 Shmyrev's Convex Program

In this section we will derive a new convex program from Eq. (12.4). In the following section, we will then show that this new convex program yields another very attractive algorithm for computing Fisher market equilibrium, by applying OMD. We introduce a change of variables to (12.4), by letting $q_j = \log p_j$ and $\gamma_i = -\log \beta_i$. Plugging these definitions into (12.4) we get

$$\begin{aligned} \min_{q,\gamma} \quad & \sum_j e^{q_j} + \sum_{i\in[n]} B_i\gamma_i \\ & q_j + \gamma_i \geq \log v_{ij}, \quad \forall i, j. \end{aligned} \tag{12.7}$$

Now we introduce Lagrangian variables b_{ij} for the constraint in (12.7) to get the following dual:

$$\max_{b\geq 0} \min_{q,\gamma} \sum_j e^{q_j} + \sum_{i\in[n]} B_i\gamma_i + \sum_{ij} b_{ij}\left[\log v_{ij} - q_j - \gamma_i\right]$$

$$= \max_{b\geq 0}\left[\sum_{ij} b_{ij}\log v_{ij} + \sum_{j\in[m]} \min_{q_j}\left[e^{q_j} - \sum_{i\in[n]} b_{ij}q_j\right] + \sum_{i\in[n]} \min_{\gamma_i} \gamma_i\left[B_i - \sum_{j\in[m]} b_{ij}\right]\right].$$

First-order optimality on γ_i shows $B_i = \sum_{j\in[m]} b_{ij}$ and first-order optimality on q_j shows $e^{q_j} = \sum_{i\in[n]} b_{ij}$. In a slight abuse of notation, we will introduce a dual variable $p_j = e^{q_j}$ (as of right now, it is not clear that this will be the same prices as the original price variables, but it turns out that this indeed holds at optimality). Putting this together we get *Shmyrev's convex program*:

$$\begin{aligned}\max_{b\geq 0} \quad & \sum_{ij} b_{ij}\log v_{ij} + \sum_{j\in[m]} \left(p_j - p_j \log p_j\right) \\ s.t. \quad & \sum_{i\in[n]} b_{ij} = p_j, \quad \forall j = 1,\dots,m, \\ & \sum_{j\in[m]} b_{ij} = B_i, \quad \forall i = 1,\dots,n.\end{aligned} \tag{12.8}$$

The variable b_{ij} can be interpreted as how much of their budget buyer i spends on a given item j. Since $\sum_{j\in[m]} p_j = \sum_{i\in[n]} B_i$ is a constant it does not affect the objective, so we may rewrite Shmyrev's CP as

$$\begin{aligned}\max_{b\geq 0} \quad & \sum_{ij} b_{ij}\log v_{ij} - \sum_{j\in[m]} p_j \log p_j \\ s.t. \quad & \sum_{i\in[n]} b_{ij} = p_j, \quad \forall j = 1,\dots,m, \\ & \sum_{j\in[m]} b_{ij} = B_i, \quad \forall i = 1,\dots,n.\end{aligned} \tag{Shmyrev}$$

12.3 Proportional Response Dynamics

We will now apply online mirror descent (OMD) to (Shmyrev). Remember that OMD makes updates according to the rule:

$$x_{t+1} = \arg\min_{x\in X} \langle \eta \nabla f_t(x), x\rangle + D(x\|x_t),$$

where $\eta > 0$ is the stepsize and $D(x\|x_t)$ is the Bregman divergence between x and x_t.

In order to instantiate OMD, we first rewrite (Shmyrev) in terms of b_{ij} only (letting $p_j(b) = \sum_i b_{ij}$) to get the objective function

$$f(b) = -\sum_{ij} b_{ij}\log v_{ij} + \sum_{j\in[m]} p_j(b)\log p_j(b) = -\sum_{ij} b_{ij}\log(v_{ij}/p_j(b)).$$

The feasible set is the set of feasible allocations of expenditures towards items:

$$X = \left\{ b \in \mathbb{R}^{n\times m}_{\geq 0} | \sum_{j\in[m]} b_{ij} = B_i, \forall i \right\}.$$

Finally, we use the distance function $d(b) = \sum_{ij} b_{ij}\log b_{ij}$ which gives $D(b\|a) = \sum_{ij} b_{ij}\log(b_{ij}/a_{ij})$

At each time t, we see the loss $f(b^t)$. The gradient is $\nabla_{ij} f(b) = 1 - \log(v_{ij}/p_j(b))$. Similar to when using the negative entropy on the simplex, the OMD update becomes:

$$\begin{aligned} b_{ij}^{t+1} &\propto b_{ij}^t \exp(-1 + \log(v_{ij}/p_j(b))) \\ &\propto b_{ij}^t \left(v_{ij}/p_j(b)\right) \\ &= \frac{1}{Z} b_{ij}^t \left(v_{ij}/p_j(b)\right), \end{aligned}$$

where Z is a normalization constant such that $\sum_{j \in [m]} b_{ij}^{t+1} = B_i$. We used a stepsize $\eta = 1$ in the above derivation. This will be justified by Lemma 12.1 below. Amazingly, OMD on (Shmyrev) using a stepsize of 1 becomes the following very natural algorithm:

- At each time t, each buyer i submits a bid vector b_i^t (the current OMD recommendation).
- Given the bids, a price $p_j^t = \sum_{i \in [n]} b_{ij}^t$ is computed for each item.
- Each buyer is given $x_{ij}^t = \frac{b_{ij}^t}{p_j^t}$ of each item.
- Each buyer submits their next bid on item j proportional to the utility they received from item j in round t:

$$b_{ij}^{t+1} = B_i \frac{x_{ij}^t v_{ij}}{\sum_{j'} x_{ij'}^t v_{ij'}}.$$

It remains to discuss the fact that we set $\eta = 1$. In earlier chapters, we saw that the uniform average of OMD iterates converges to zero average regret at a rate of $O(1/\sqrt{T})$, when using a stepsize proportional to the inverse of the largest observed dual norm of gradients. However, our objective f does not admit such a bound: the gradient for i, j goes to infinity as $p_j(b)$ tends to zero. Thus based on our existing framework for OMD we are not even guaranteed a bound on regret. However, it turns out that one can show the following "1-Lipschitz" condition, by measuring Lipschitzness relative to D, rather than the usual Euclidean distance:

Lemma 12.1 *For all $a, b \in S$,*

$$f(b) \leq f(a) + \langle \nabla f(a), b - a \rangle + D(b \| a), \quad \forall b, a \in X.$$

This inequality is a generalized Lipschitz condition where we replace the squared ℓ_2 norm $\|a - b\|_2^2$ with our Bregman divergence D (this is analogous to how OMD itself generalized projected gradient descent by changing the distance function).

To show this inequality, we will need the fact that the Bregman divergence $D(b\|a)$ is convex in both arguments for $b, a \in \mathbb{R}^{n\times m}_{++}$. To see that convexity holds, one can expand $D(b\|a) = \sum_{ij} b_{ij} \log(b_{ij}/a_{ij})$ and note that taking a sum preserves convexity. At that point, we only need to check convexity of the function $h(t, x) = t\log(t/x) = -t\log(x/t)$, which is simply the perspective of $-\log(x)$ with respect to t. Taking perspectives is known to preserve convexity, and the negative log is of course convex.

Proof The proof of the inequality can be split into two parts. First, it can be observed that the difference between $f(b)$ and its linearization at a is the Bregman divergence $D(p(b)\|p(a))$:

$$\begin{aligned}
&f(b) - f(a) - \langle \nabla f(a), b - a\rangle \\
&= -\sum_{ij} b_{ij}\log(v_{ij}/p_j(b)) + \sum_{ij} a_{ij}\log(v_{ij}/p_j(a)) \\
&\quad - \sum_{ij}\left(1 - \log(v_{ij}/p_j(a))\right)(b_{ij} - a_{ij}) \\
&= -\sum_{ij} b_{ij}\log(v_{ij}/p_j(b)) + \sum_{ij} b_{ij}\log(v_{ij}/p_j(a)) - \sum_{ij}(b_{ij} - a_{ij}) \\
&= \sum_{ij} b_{ij}\log(p_j(b)/p_j(a)) - \sum_{ij}(b_{ij} - a_{ij}) \\
&= \sum_{ij} b_{ij}\log(p_j(b)/p_j(a)) \quad ;\text{since } \|a\|_1 = \|b\|_1 = \sum_{i\in[n]} B_i \\
&= \sum_{j} p_j(b)\log(p_j(b)/p_j(a)) \quad ;\text{since } p_j(b) = \sum_{i\in[n]} b_{ij} \\
&= D(p(b)\|p(a)).
\end{aligned}$$

Secondly, we can bound $D(p(b)\|p(a))$ as follows (where $h(t, x) = t\log(t/x)$)

$$\begin{aligned}
D(p(b)\|p(a)) &= n\sum_{j\in[m]} \frac{1}{n} h(p_j(b), p_j(a)) \\
&= n\sum_{j\in[m]} h\left(\frac{1}{n}p_j(b), \frac{1}{n}p_j(a)\right) \\
&\leq n\sum_{j\in[m]} \frac{1}{n}\sum_{i\in[n]} h\left(b_{ij}, a_{ij}\right) \\
&= D(b\|a).
\end{aligned}$$

Putting together the two bounds we get Lemma 12.1. □

Using the Lipschitz-like condition on f, one can show a stronger statement when running OMD on a static objective f (which means that it is the same as running normal mirror descent):

Theorem 12.2 *The OMD iterates with $\eta = 1$ converge at the rate:*

$$f(b^t) - f(b^*) \leq \frac{\log nm}{t}.$$

This holds for any convex and differentiable f and D satisfying 12.1.

Note two very nice properties here: the convergence rate is improved by a factor of $\sqrt{t}$, and the iterates themselves converge, with no need for averaging. We won't prove the above theorem here, but it holds for any convex minimization problem that satisfies the relative Lipschitz condition in Lemma 12.1.

12.4 Abstraction Methods

So far we have described a scalable first-order method for computing market equilibrium. Still, this algorithm makes a number of assumptions that may not hold in practice. First, the size of an iterate b^t is nm; if both are on the order of 100,000 then writing down an iterate using 64-bit floats requires about 80 GB of memory. For an application such as an Internet advertising market we might expect the number of buyers n, and especially the number of items m, to be even larger than that. Thus, we may need to find a way to abstract that market down to some manageable size where we can at least hope to write down iterates. Secondly, in practice we may not have access to all v_{ij}. Instead, we may only have samples from v_{ij}, and we need to somehow infer the remaining valuations. We now move to considering abstraction methods, which will allow us to deal with both of the above issues.

For the purposes of abstraction, it will be useful to think of the set of valuations v_{ij} as a matrix V, where the i'th row corresponds to the valuation vector of buyer i. We will be interested in what happens if we compute a market equilibrium using some valuation matrix $\tilde{V} \neq V$, where $\tilde{V}$ would typically be obtained from some abstraction method that generates a smaller $\tilde{V}$ (for some appropriate measure of smaller, e.g. due to a low-rank approximation). Can we say anything about how "close" to market equilibrium we are in terms of the original V, for example if $\|\tilde{V} - V\|_F$ is small? We first describe two reasons that we might compute a market equilibrium for $\tilde{V}$ rather than V:

(i) *Low-rank markets*: When there are missing valuations, we need to somehow impute the missing values. Of course, if there is no relationship between the entries of V that we observed, and those that are missing, then we have no hope of recovering V. However, in practice this is typically not the case. In practice, the valuations are often assumed to (approximately) belong to some low-dimensional space. A popular model is to assume that the valuations are *low rank*, meaning that every buyer i has some d-dimensional vector ϕ_i, every good j has some d-dimensional vector ψ_j, and the valuation of buyer i for good j is $\tilde{v}_{ij} = \langle \phi_i, \psi_j \rangle$. One may interpret this model as every item having some associated set of d *features*, with ψ_j describing the value for each feature, and ϕ_i describes the value that i places on each feature. In a low-rank model d is expected to be much smaller than $\min(n, m)$, meaning that V is far from full rank. If the real valuations do not have this structure, but are approximately rank d (meaning that the sum of singular values $\sum_{k=d+1}^{\min(n,m)} \sigma_k$ is small), then $\tilde{V}$ will be close to V.

This model can also be motivated via the singular-value decomposition (SVD). Assume that we wish to solve the following problem:

$$\min_{\tilde{V}} \sum_{ij} (v_{ij} - \tilde{v}_{ij})^2 = \|V - \tilde{V}\|_F^2$$
$$s.t.\ \operatorname{rank}(\tilde{V}) \leq d.$$

The optimal solution to this problem can be found easily via the SVD: Letting $\sigma_1, \ldots, \sigma_d$ be the first d singular values of V, and $\bar{u}_1, \ldots, \bar{u}_d$ the first left singular vectors, and $\bar{v}_1, \ldots, \bar{v}_d$ the first right singular vectors, the optimal solution is

$$\tilde{V} = \sum_{k=1}^{d} \sigma_k \bar{u}_k \bar{v}_k^T.$$

If the remaining singular values $\sigma_{k+1}, \ldots$ are small relative to the first k singular values, then this model captures most of the valuation structure.

(ii) *Representative Markets*: We may wish to try to generate a smaller set of representative buyers, where each original buyer i maps to some particular representative buyer $r(i)$. Similarly, we may wish to generate representative goods that correspond to many non-identical but similar goods from the original market. In practice these representative buyers and goods would typically be generated via clustering techniques. In this case, our approximate valuation matrix $\tilde{V}$ has as row i the valuation vector of the representative buyer $r(i)$. This means that all i, i' such that $r(i) = r(i')$ have the same valuation vector in $\tilde{V}$, and thus they can be treated as a single buyer for equilibrium-computation purposes. The same grouping can also be applied to the goods. If the number of buyers and goods is reduced by a factor of 10, then the resulting mathematical program is reduced by a factor of 10^2, since we have $n \times m$ variables.

12.4.1 Measuring Solution Quality

We now analyze what happens when we compute a market equilibrium under $\tilde{V}$ rather than V. Throughout this section we will let $(\tilde{x}, \tilde{p})$ be a market equilibrium for $\tilde{V}$. We will use the error matrix $\Delta V = V - \tilde{V}$ to quantify the solution quality, and we will measure the size of ΔV using the $\ell_1 - \ell_\infty$ matrix norm:

$$\|\Delta V\|_{1,\infty} = \max_i \|\Delta v_i\|_1.$$

We will also use the norm of the error vector for an individual buyer $\|\Delta v_i\|_1 = \|v_i - \tilde{v}_i\|_1$.

A very useful property is that under linear utilities, the change in utility when going from v_i to $\tilde{v}_i$ is linear in Δv_i.

Proposition 12.3 *If* $\langle \tilde{v}_i, x_i\rangle + \epsilon \geq \langle \tilde{v}_i, x_i'\rangle$ *then* $\langle v_i, x_i\rangle + \epsilon + \|\Delta v_i\|_1 \geq \langle v_i, x_i'\rangle$

Proof We have

$$\begin{aligned} &\langle \tilde{v}_i, x_i\rangle + \epsilon \geq \langle \tilde{v}_i, x_i'\rangle \\ \Leftrightarrow\ &\langle v_i - \Delta v_i, x_i\rangle + \epsilon \geq \langle v_i + \Delta v_i, x_i'\rangle \\ \Leftrightarrow\ &\langle v_i, x_i\rangle + \langle \Delta v_i, x_i' - x_i\rangle + \epsilon \geq \langle v_i, x_i'\rangle. \end{aligned}$$

Now the proposition follows by $\langle \Delta v_i, x_i' - x_i\rangle \leq \|\Delta v_i\|_1$. □

This proposition can be used to immediately derive bounds on envy, proportionality, and regret (how far each buyer is from achieving the utility of their demand bundle). For example, we know that under $\tilde{V}$, each buyer i has no envy towards any other buyer k: $\langle \tilde{v}_i, \tilde{x}_i\rangle \geq \langle \tilde{v}_i, \tilde{x}_k\rangle$. By Proposition 12.3 each buyer i has envy at most $\|\Delta v_i\|_1$ under V when using $(\tilde{x}, \tilde{p})$. All envies are thus bounded by $\|\Delta V\|_{1,\infty}$. Regret and proportionality is bounded similarly using guaranteed inequalities under $\tilde{V}$.

Market equilibrium also guarantees Pareto optimality. Can we give any meaningful guarantees on how much social welfare improves under Pareto-improving allocations for $\tilde{V}$? Unfortunately the answer to that is no, as the following example of real and abstracted matrices shows:

$$V = \begin{bmatrix} 1 & \epsilon & \epsilon \\ 0 & 1 & \epsilon \end{bmatrix}, \quad \tilde{V} = \begin{bmatrix} 1 & \epsilon & 0 \\ 0 & 1 & \epsilon \end{bmatrix}.$$

If we set $B_1 = B_2 = 1$, then for supply-aware market equilibrium, we end up with competition only on item 2, and we get prices $\tilde{p} = (0, 2, 0)$ and allocation $\tilde{x}_1 = (1, 0.5, 0), \tilde{x}_2 = (0, 0.5, 1)$. Under V this is a terrible allocation, and we can Pareto improve by using $x_1 = (1, 0, 0.5), x_2 = (0, 1, 0.5)$, which increases overall social welfare by $\frac{1}{2} - \epsilon$, in spite of $\|\Delta V\|_1 = \epsilon$.

On the other hand, we can show that under any Pareto-improving allocation, some buyer i improves by at most $\|\Delta V\|_{1,\infty}$. To see this, note that for any Pareto improving allocation x, under $\tilde{V}$ there existed at least one buyer i such that $\langle \tilde{v}_i, \tilde{x}_i - x_i \rangle \geq 0$, and so this buyer must improve by at most $\|\Delta v_i\|_1$ under V.

12.5 Historical Notes

The Shmyrev convex program was given by Shmyrev (2009). The observation that the Shmyrev convex program is related to EG via duality and change of variables was by Cole et al. (2017). The original proportional response dynamics were given by Wu and Zhang (2007), and was shown to be effective for BitTorrent sharing dynamics by Levin et al. (2008). The relationship of proportional response dynamics to Shmyrev's convex program and mirror descent were given by Birnbaum et al. (2011). For rules on convexity-preserving operations, see Boyd and Vandenberghe (2004).

There is a long history of first-order algorithms for computing market equilibrium in various Fisher-market models. We focused on proportional response dynamics, which have particularly strong numerical performance.

There is a rich literature on various iterative approaches to computing market equilibrium in Fisher markets. One can apply first-order methods or regret-minimization approaches to the Lagrangian of EG directly, which was done in Kroer et al. (2019) and Gao et al. (2021a). Classical projected gradient descent applied directly on EG achieves linear convergence (Gao and Kroer, 2020). There is a large optimization and computer science literature on discrete-time tâtonnement dynamics (Cole and Fleischer, 2008; Cheung et al., 2012, 2019; Nan et al., 2024). In the economics literature, tâtonnement was initially studied in continuous time, see e.g. Mas-Colell et al. (1995). There is also a literature deriving auction-like algorithms, which can similarly sometimes be viewed as instantiations of gradient descent and related algorithms (Bei et al., 2019; Nesterov and Shikhman, 2018; Gao et al., 2021b; Liao et al., 2022; Yang et al., 2024).

The material on abstracting large market equilibrium problems is from Kroer et al. (2019). A brief introduction to the broader idea of low-rank models can be found in Udell and Townsend (2019). Udell et al. (2016) gives a more thorough exposition and describes more general model types. Low-rank market equilibrium models were also studied in Kroer and Peysakhovich (2019), where it is shown that large low-rank markets enjoy a number of properties not satisfied by small-scale markets.

Further reading.

Cole et al. (2017) is a good reference for how to derive the dual of the Eisenberg-Gale convex program, and they also show how to extend those results to a variety of settings such as for quasi-linear utilities. Birnbaum et al. (2011) is a good reference for the relationship between proportional response dynamics and Shmyrev's convex program, and they give a variety of useful inequalities for analyzing proportional response dynamics.

13
Fair Allocation with Indivisible Goods

In this chapter we study the problem of fair allocation when the items are indivisible. This setting presents a number of challenges that were not present in the divisible case.

It is obviously an important setting in practice. For example, the website `http://www.spliddit.org/` allows users to fairly split estates, financial assets, toys, or other goods. Another important application is that of fairly allocating course seats to students. This setting is even more intricate, because valuations in that setting are combinatorial. In order to design suitable mechanisms for fairly dividing discrete goods, we will need to reevaluate our fairness concepts.

The setup is similar to Chapter 11. We have a set of m indivisible goods that we wish to divide among n agents. Each good has supply 1. We will denote the bundle of goods given to agent i as x_i, where x_{ij} is the amount of good j that is allocated to buyer i. The set of feasible allocations is then $\{x \mid \sum_{i \in [n]} x_{ij} \leq 1 \; \forall j \in [m], x_{ij} \in \{0, 1\}\}$

Unless otherwise specified, each agent is assumed to have a linear utility function $u_i(x_i) = v_i^\top x_i$ denoting how much they like the bundle x_i.

13.1 Fair Allocation

In the case of indivisible items, several of our fairness properties become much harder to achieve. We will assume that we are required to construct a Pareto-optimal allocation.

Proportional fairness does not even make sense anymore: it rested on the idea of assigning each agent their fractional share $\frac{1}{n}$ of each item. There is however, a suitable generalization of proportionality that does make sense for the indivisible case: the *maximin share (MMS) guarantee*. For agent i, their

MMS guarantee is the value they would get if they get to divide the items up into n bundles, but were then required to take the worst bundle. Formally:

$$\begin{aligned} \text{MMS}_i := \max_{x \geq 0} \ & \min_j u_i(x_j) \\ \text{s.t.} & \sum_{i \in [n]} x_{ij} \leq 1, \forall j \\ & x_{ij} \in \{0, 1\}, \forall i, j. \end{aligned}$$

For the divisible setting, the value of the above optimization problem becomes the buyer's utility of the proportional allocation. We say that an allocation x is an MMS allocation if every agent i receives at least their MMS share, i.e. utility $u_i(x_i) \geq \text{MMS}_i$. In the case of 2 agents, an MMS allocation always exists. As an exercise, you might try to come up with an algorithm for finding such an allocation.[1] In the case of 3 or more agents, an MMS allocation may not exist. The counterexample is very involved, so we won't cover it here.

Theorem 13.1 *For $n \geq 3$ agents, there exist additive valuations for which an MMS allocation does not exist. However, an allocation x such that each agent receives utility at least $u_i(x) \geq \frac{3}{4}\text{MMS}_i$ always exists.*

The original Spliddit algorithm for dividing goods worked as follows: first, compute (e.g. via integer programming) the $\alpha \in [0, 1]$ such that every agent can be guaranteed an α fraction of their MMS guarantee (this always ends up being $\alpha = 1$ in practice). Then, an allocation that maximizes social welfare subject to supply feasibility and the constraints $u_i(x_i) \geq \alpha\,\text{MMS}_i$ for all $i \in [n]$ is computed. However, this can lead to some weird results.

Example 13.2 Three agents each have valuation 1 for 5 items. The MMS guarantee is 1 for each agent. But now the social welfare-maximizing solution can allocate three items to agent 1, and 1 item each to agents 2 and 3. A more fair solution would be to allocate 2 items to 2 agents, and 1 item to the last agent. One way to quantify how unfair the 3/1/1 solution is, as compared to the 2/2/1 solution, is that envy is twice as high in the 3/1/1/ solution.

With the above motivation, let us consider envy in the discrete setting. It is easy to see that we generally won't be able to get envy-free solutions if we are required to assign all items. Consider 2 agents splitting an inheritance: a house worth \$500k, a car worth \$10k, and a jewelry set worth \$5k. Since we have

[1] Solution: compute one of the solutions to agent 1's MMS computation problem. Then let agent 2 choose their favorite bundle, and give the other bundle to agent 1. Agent 1 clearly receives their MMS guarantee, or better. Agent 2 also does: their MMS guarantee is at most $\frac{1}{2}\|v_2\|_1$, and here they receive utility of at least $\frac{1}{2}\|v_2\|_1$.

to give the house to a single agent, the other agent is guaranteed to have envy. Instead, we will consider a relaxed notion of envy:

Definition 13.3 An allocation x is *envy-free up to one good* (EF1) if for every pair of agents i, k, there exists an item j such that $x_{kj} = 1$ and $u_i(x_i) \geq u_i(x_k - e_j)$, where e_j is the j'th basis vector.

Intuitively, this definition says that for any pair of agents such that one agent envies the other, we only need to remove one item from the bundle in order to remove the envy. Requiring EF1 would have forced us to use the 2/2/1 allocation in Example 13.2.

For linear utilities, an EF1 allocation is easily found (if we disregard Pareto optimality). As an exercise, come up with an algorithm for computing an EF1 allocation for linear valuations[2] In fact, EF1 allocations can be computed in polynomial time for any monotone set of utility functions (meaning that if $x_i \geq x_i'$ then $u_i(x_i) \geq u_i(x_i')$), using an algorithm known as envy-cycle-elimination.

However, ideally we would like to come up with an algorithm that gives us EF1 as well as Pareto efficiency. To achieve this, we will consider the product of utilities, which we saw previously in the Eisenberg-Gale program. This product is also called the *Nash welfare* of an allocation:

$$NW(x) = \prod_i u_i(x_i).$$

The *max Nash welfare* (MNW) solution picks an allocation that maximizes $NW(x)$:

$$\begin{aligned} \max_x \; & \prod_i u_i(x_i) \\ s.t. \; & \sum_{i \in [n]} x_{ij} \leq 1, \forall j \\ & x_{ij} \in \{0, 1\}, \forall i, j. \end{aligned}$$

Note that here we have to worry about the degenerate case where $NW(x) = 0$ for *all* x, meaning that it is impossible to give strictly positive utility to all agents. We will assume that there exists x such that $NW(x) > 0$. If this does not hold, typically one seeks a solution that maximizes the number of agents with strictly

[2] This is achieved by the round-robin algorithm: simply have agents take turns picking their favorite item. It is easy to see that EF1 is an invariant of the partial allocations resulting from this process.

positive utility, and then the largest NW achievable among subsets of that size is chosen.

The MNW solution turns out to achieve both Pareto optimality (otherwise we contradict optimality), and EF1:

Theorem 13.4 *The MNW solution for linear utilities is EF1.*

Proof Let x be the MNW solution. Say for contradiction that agent i envies agent k by more than one good. Let j be the item allocated to agent k that minimizes the ratio $\frac{v_{kj}}{v_{ij}}$. Let x' be the same allocation as x, except we give item j to agent i instead, i.e. $x'_{ij} = 1, x'_{kj} = 0$. The proof is by showing that $NW(x') > NW(x)$, which contradicts optimality of x for the MNW problem.

Using the linearity of utilities we have $u_i(x'_i) = u_i(x_i) + v_{ij}$ and $u_k(x'_k) = u_k(x_k) - v_{kj}$. Every other utility stays the same. Now we have

$$
\begin{aligned}
& NW(x') > NW(x) \\
\Leftrightarrow & \left[u_i(x_i) + v_{ij}\right] \cdot \left[u_k(x_k) - v_{kj}\right] > u_i(x_i)u_k(x_k) \\
\Leftrightarrow & \left[u_i(x_i) + v_{ij}\right] u_k(x_k) > u_i(x_i)u_k(x_k) + \left[u_i(x_i) + v_{ij}\right] v_{kj} && (13.1) \\
\Leftrightarrow & v_{ij}u_k(x_k) > \left[u_i(x_i) + v_{ij}\right] v_{kj} && (13.2) \\
\Leftrightarrow & u_k(x_k) > \frac{v_{kj}}{v_{ij}} \left[u_i(x_i) + v_{ij}\right]. && (13.3)
\end{aligned}
$$

By how we chose j we have (see Eq. (A.5)):

$$\frac{v_{kj}}{v_{ij}} \le \frac{\sum_{j' \in x_k} v_{kj'}}{\sum_{j' \in x_k} v_{ij'}} \le \frac{u_k(x_k)}{u_i(x_k)},$$

and by the envy property we have

$$u_i(x_i) + v_{ij} < u_i(x_k).$$

Now we can multiply together the last two inequalities to get (13.3). □

The MNW solution also turns out to give a guarantee on MMS, but not a very strong one: every agent is guaranteed to get $\frac{2}{1+\sqrt{4n-3}}$ of their MMS guarantee, and this bound is tight. Luckily, in practice the MNW solution seems to fare much better. On Spliddit data, the following ratios are achieved. In the table below are shown the MMS approximation ratios across 1281 “divide goods” instances submitted to the Spliddit website for fairly allocating goods

In over 95% of the instances every player receives their full MMS guarantee.

MMS approximation ratio	[0.75, 0.8)	[0.8, 0.9)	[0.9, 1)	1
% of instances in interval	0.16%	0.7%	3.51%	95.63%

Table 13.1 *MMS approximation ratios for Spliddit instances.*

13.2 Computing Discrete Max Nash Welfare

13.2.1 Complexity

Solving the MNW problem is generally not easy. In fact, the problem turns out to be not only NP-hard, but NP-hard to approximate within a factor $\mu \approx 1.00008$ (the best current algorithm achieves an approximation factor of 1.45, so the gap between 1.00008 and 1.45 is open).

The reduction is based on the vertex-cover problem on 3-regular graphs, which is NP-hard to approximate within a factor ≈ 1.01. The vertex cover problem asks: given a graph G, find the smallest set of vertices such that every edge is incident to at least one of them. The decision version of the problem is: "Given G and integer k, does G contain a vertex cover of size $\leq k$?" A 3-regular graph is a graph where each vertex has degree 3.

The proof is not particularly illuminating, so we will skip it here. However, let's see a quick way to prove a simpler statement: that the problem is NP-hard even for 2 players with identical linear valuations. The hardness is by reduction from the PARTITION problem, which is a well-known NP-hard problem.

Definition 13.5 PARTITION problem: you are given a multiset of integers $S = \{s_1, \ldots, s_m\}$ (potentially with duplicates), and your task is to figure out if there is a way to partition S into two sets S_1, S_2 such that $\sum_{i \in S_1} s_i = \sum_{i \in S_2} s_i$.

Given a PARTITION instance with multiset $S = \{s_1, s_2, \ldots, s_m\}$, we can construct an MNW instance as follows: we create two agents and m items. Both agents have value s_j for item j (and thus identical valuations). Now, by the AM-GM inequality (see Eq. (A.2)) there exists a correct partitioning if and only if the MNW allocation has value $(\frac{1}{2} \sum_{j \in [m]} s_j)^2$.

This result can be extended to show strong NP-hardness by considering the k-EQUAL-SUM-SUBSET problem: given a multiset $\mathcal{S}$ of $x_1, \ldots, x_n$ positive integers, are there k nonempty disjoint subsets $S_1, \ldots, S_k \subset \mathcal{S}$ such that $sum(S_1) = \ldots = sum(S_k)$. The exact same reduction as before works, but with k agents rather than two.

13.2.2 Algorithms

Given these computational complexity problems, how should we compute an MNW allocation in practice?

We present two approaches here. First, we can take the log of the objective, to get a concave function. After taking logs, we get the following mixed-integer exponential-cone program:

$$\begin{aligned} \max \sum_{i\in[n]} & \log u_i \\ \text{s.t. } u_i &\le \langle v_i, x_i\rangle, && \forall i \in [n] \\ \sum_{i\in[n]} x_{ij} &\le 1, && \forall j \in [m] \\ x_{ij} &\in \{0,1\}, && \forall i \in [n], j \in [m]. \end{aligned} \tag{13.4}$$

This is simply the discrete version of the Eisenberg-Gale convex program. One approach is to solve this problem directly, e.g. using Mosek.

Alternatively, we can impose some additional structure on the valuation space: if we assume that all valuations are integer-valued, then we know that $u_i(x_i)$ will take on some integer value in the range 0 to $\|v_i\|_1$. In that case, we can add a variable w_i for each agent i, and use either (1) the linearization of the log at each integer value, or (2) the linear function from the line segment $(\log k, k), (\log(k+1), k+1)$, as upper bounds on w_i. This gives $\frac{1}{2}\|v_i\|_1$ constraints for each i using the line segment approach (the linearization uses twice as many constraints), but ensures that w_i is equal to $\log\langle v_i, x_i\rangle$ for all integer-valued $\langle v_i, x_i\rangle$. Using the line segment approach gives the following mixed-integer linear program (MILP):

$$\begin{aligned} \max \sum_{i\in[n]} & w_i \\ \text{s.t. } w_i &\le \log k + \frac{\log(k+1)}{\log k}\big(\langle v_i, x_i\rangle - k\big), && \forall i \in [n], k = 1, 3, \ldots, \|v_i\|_1 \\ \sum_{j\in[m]} v_{ij}x_{ij} &\ge 1, && \forall i \in [n] \\ \sum_{i\in[n]} x_{ij} &\le 1, && \forall j \in [m] \\ x_{ij} &\in \{0,1\}, && \forall i \in [n], j \in [m]. \end{aligned} \tag{13.5}$$

These two mixed-integer programs both have some drawbacks: For the mixed-integer exponential-cone program in Eq. (13.4), we must resort to much less mature technology than for mixed-integer linear programs. On the other hand, the discrete EG program is reasonably compact: the program is roughly the size of a solution. For the MILP in Eq. (13.5), the good news is that MILP technology is quite mature, and so we might expect this to solve quickly. On the other hand, adding $n \times \|v_i\|_1$ additional constraints can be quite a lot, and could lead to slow LP solves as part of the branch-and-bound procedure.

Figure 13.1 shows the performance of the two approaches.

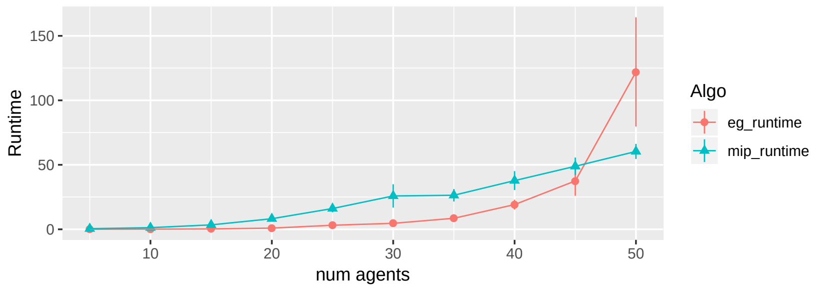

Figure 13.1 Plot showing the runtime of discrete Eisenberg-Gale and the MILP approach.

13.3 Fair Allocation with Combinatorial Utilities

Recall that for the setting of indivisible goods, a market equilibrium is not guaranteed to exist. Moreover, envy-free allocations are also not guaranteed to exist. In this section we will see how to recover existence by considering an appropriate notion of approximate market equilibria, which will be guaranteed to exist and yield approximate envy freeness. Our setup will allow for a very broad class of combinatorial utilities, and we will use this to model the allocation of course seats to students.

Specifically, we will look at a generalization of the *competitive equilibrium from equal incomes* (CEEI) allocation mechanism. Since a market equilibrium is not guaranteed to exist for equal budgets, we will instead look at *approximate CEEI* (A-CEEI). In A-CEEI the idea is to relax two parts of CEEI: (1) we give agents approximately equal, rather than exactly equal, budgets, and (2) we only clear the market approximately. Let's see how this works with an example.

Example 13.6 Two students are trying to register for four courses, where each course has a single seat left, and each student can take at most two courses. The courses are: machine learning (ML), statistics (STAT), algorithms (ALGO), and real analysis (RA). Both students appreciate the finer things in life, so they prefer any bundle that includes real analysis over any bundle that does not. More generally, both students' preferences are such that they rank the courses in the order RA > ML > STAT > ALGO. Their preferences over bundles of two courses are to always prefer the bundle with the highest-ranked course in it, breaking ties using the lower-ranked course.

Clearly, if budgets are equal we cannot hope to price these items in a way that clears the market, since both students will always want the bundle with the real analysis course int it if they can afford it. But if we instead give student 1 a budget of 1.2 and student 2 a budget of 1, then we can set the prices as follows, RA: 1.1, ML: 0.8, STAT: 0.2, ALGO: 0.1. Now student 1 wishes to buy (RA, ALGO) for a total price of 1.2, and student 2 wishes to buy (ML, STAT) for a total price of 1.

As long as we decide the budget perturbations in a randomized way, the allocation in Example 13.6 is in some sense fair in expectation, and furthermore we might hope that the budget perturbations are small enough that for instances with more than four items, things look even fairer. Note that the allocation we found satisfies both EF1 and the MMS guarantee. The example also achieves Pareto optimality, though we will in general only guarantee approximate Pareto optimality for A-CEEI.

13.3.1 Approximate CEEI

We will describe the problem in the context of matching students to seats in courses. This setup is used in the *Course Match* software, which is used for matching students to course seats in a variety of business schools in the US and Canada. There is a set of m courses, and each course j has some integer capacity s_j. There is a set of n students. Each student has a set $\Psi_i \subseteq 2^m$ of feasible subsets of courses that they may be allocated, with each bundle containing at most $k \leq m$ courses (note that this assumes that each student can only consume one unit of a good, even if $s_j > 1$; this is of course reasonable in course allocation, but not for all applications). The set Ψ_i encodes both scheduling constraints such as mutually exclusive courses that are taught in the same time slot, as well as constraints specific to the student such as whether they satisfy course prerequisites.

So far in the book we have only worked with cardinal preferences, where each agent i has a numerical value specifying how much they like a given outcome. However, the Approximate CEEI setting is historically studied as an *ordinal* setting. In an ordinal setting we are not given utility values for outcomes, but are instead given an *ordering* over outcomes. We gave ordinal rankings in the two-student two-course Example 13.6. Informally, given two bundles of courses x_i, x_i', the ordering tells us which of the two bundles student i prefers. If the reader is not familiar with ordinal settings, then it is fine to think of these orderings as simply being the result of the student having some underlying cardinal utility that generates the pairwise preferences.

The preferences of student i are assumed to be given as a complete and transitive ordinal preference ordering $\succcurlyeq_i$ over Ψ_i. Completeness simply means that for all schedules $x, x' \in \Psi_i$, $x \succcurlyeq_i x'$, $x' \succcurlyeq_i x$, or both. Transitivity means that if $x \succcurlyeq_i x'$ and $x' \succcurlyeq_i x''$ then $x \succcurlyeq_i x''$.

Example 13.7 In Course Match, the representation of $\succcurlyeq_i$ is as follows: the set of feasible schedules Ψ_i is taken as given. Then, student i ranks each course on an integer scale from $0 - 100$, and is additionally allowed to specify pairwise penalties or bonuses in $[-200, -199, \dots, 199, 200]$ for being assigned a given pair of courses. Thus, the students' utility for a bundle is the sum of their values for the individual courses in the bundles, plus the sum of penalties and bonuses for all pairs of courses in the bundle. This induces a complete and transitive preference ordering.

Given a set of prices p for each course, a vector x_i^* is in the demand set for student i if

$$x_i^* \in D_i(p) = \arg\max_{\succcurlyeq_i} \{x_i \in \Psi_i : \langle x_i, p \rangle \leq B_i\}.$$

This maximum is well-defined because we assumed that preferences are complete. In the actual Course Match implementation, $\succcurlyeq_i$ is represented numerically by the utility function described in Example 13.7 for each student, but the A-CEEI theory works for the more general case of ordinal preferences.

Since we have existence issues (these arise both from indivisibility as seen earlier, but also from the very general preference orderings allowed), we resort to an approximation to CEEI:

Definition 13.8 An allocation x, prices p, and budgets B constitute an (α, β)-CEEI if:

(i) Demands are met: $x_i \in D_i$ for all $i \in [n]$.
(ii) α-approximate market clearance: $\|z\|_2 \leq \alpha$, where $z \in \mathbb{R}^m_{\geq 0}$ is defined as $z_j = \sum_{i \in [n]} x_{ij} - s_j$ if $p_j > 0$, and $z_j = \max(\sum_{i \in [n]} x_{ij} - s_j, 0)$ if $p_j = 0$.
(iii) β-approximately equal budgets: $B_i \in [1, 1 + \beta]$ for all i.

The first condition in (α, β)-CEEI simply says that each student i buys an item in their demand set. The second condition says that supply constraints are approximately satisfied, and courses with strictly positive price are approximately allocated at their supply. The third constraint says that all budgets are almost the same, up to a difference of β.

The main theorem regarding (α, β)-CEEI is that they are guaranteed to exist:

Theorem 13.9 *Let $\sigma = \min(2k, m)$. For any $\beta > 0$, there exists a $(\sqrt{\sigma m}/2, \beta)$-CEEI. Moreover, given budgets $B \in [1, 1+\beta]^n$ and any $\epsilon > 0$, there exists a $(\sqrt{\sigma m}/2, \beta)$-CEEI using budgets B^* such that $\|B^* - B\|_\infty \leq \epsilon$.*

One major concern with this result is that we are not quite guaranteed a feasible solution. In general the allocation may oversubscribe some courses, though the oversubscription vector z has bounded ℓ_2 norm. In practice, the bound is relatively modest: First, the bound $\sqrt{\sigma m}/2$ does not grow with the number of agents or number of course seats. Second, in practice students take at most a modest number of courses per semester among a reasonably-small number of courses offered (an example given in the literature is that students take $k = 5$ courses out of 50 courses total at Harvard's MBA program), thus yielding a bound of roughly 11. Technically a single course could be oversubscribed by 11 students, but in practice we expect this to be smoothed out reasonably across many courses.

The proof of the existence theorem is rather involved and relies on smoothing out the market in order to invoke fixed-point theorems. Here we give some intuition for the role that each approximation plays.

As in other discontinuous settings, the main difficulty for existence without approximation is the discontinuity of student demands with respect to price. However, in the course match setting, $\sqrt{\sigma}$ is an upper bound on the discontinuity of the demand of any single agent. To see this, note that a demand x_i has at most k entries set to 1, and so a student can at most drop all courses from x_i and switch to k new courses under their new demand x_i'. At the same time, there's only m courses total, so the change is bounded by $\min(2k, m)$, and thus $\|x_i - x_i'\|_2 \leq \sqrt{\sigma}$.

The second discontinuity issue is large discontinuous aggregate changes in demand from the students. When budgets are the same, as in standard CEEI, the demand discontinuity for different students may occur at the same point in the space of prices. Thus, if this happens, aggregate discontinuity may be on the order of $n\sigma$. With distinct budgets, it becomes possible to change a single student's demand without changing those of other students. For each bundle x, we may think of the hyperplane $H(i, x) = \{p : \langle p, x \rangle \leq B_i\}$ which denotes the boundary between two halfspaces in the price space: those where student i can afford x, and those where i cannot afford x. By having each budget distinct, one can show that in a generic sense, at most m hyperplanes can intersect at any particular point in price space. This implies that aggregate demand changes by at most σm.

The remainder of the proof is concerned with smoothing out the aggregate demands so that a fixed-point existence theorem can be applied to show existence.

Fairness and Optimality Properties of A-CEEI

Since we are only approximately clearing the market, we do not get Pareto optimality. However, it is possible to show that if we construct a modified market where $\tilde{s}_j = s_j - z_j$, then we have Pareto optimality in that market. Thus, any Pareto-improving allocation must utilize unused supply, which can potentially be used to bound the inefficiency once more structure is imposed on utilities.

Crucially, (α, β)-CEEI does guarantee some fairness properties. If we select $\beta \leq \frac{1}{k-1}$, then EF1 is guaranteed in any (α, β)-CEEI. Furthermore, there exists β small enough such that each student is also guaranteed to receive their $(n+1)$-MMS share, which is their utility if they were forced to partition the items into $n + 1$ bundles and take the worst one.

13.3.2 Computing A-CEEI

In general computing an A-CEEI is PPAD complete. This is the same class of problem that general-sum Nash equilibrium falls in. It is conjectured to require exponential time in the worst case, and thus we cannot hope to have nice scalable algorithms like we had for the divisible case.

In practice, A-CEEI is computed using local search. A *tabu search* is used on the space of prices. This works as follows:

(i) A price vector is generated randomly
(ii) A set of "neighbors" are generated using two different generation approaches:
  - "Price gradient:" all the demands under the current prices are added up, and the excess demand vector is treated as a gradient. Then, 20 different stepsizes are tried along the price gradient
  - A single item has its price changed, and all other prices are kept the same. The new price on the chosen item is set high enough to stop it from being oversubscribed, or low enough to stop being undersubscribed. A neighbor is generated for each over or undersubscribed item

(iii) The best neighbor (among the ones generating a previously-unseen allocation) is selected as the next price vector, and the procedure repeats from step 2 (unless the last 5 iterations yielded no improving prices, in which case the local search stops)
(iv) Finally, step 1 is repeated with a new random price vector. This repeats until a time limit is reached

In practice this procedure generates an A-CEEI solution with significantly better α and β values than the theory predicts. After an A-CEEI has been generated, additional heuristics are implemented in order to force the solution to not have oversubscription.

13.4 Historical Notes

The maximin share was introduced by Budish (2011). The results on nonexistence of MMS allocation, and an approximation guarantee of $\frac{2}{3}$ were given by Kurokawa et al. (2018). The approximation guarantee was improved to $\frac{3}{4}$ by Ghodsi et al. (2018). The application of MNW to fair division was proposed by Caragiannis et al. (2016).

The envy freeness up to one good (EF1) notion was implicitly introduced by Lipton et al. (2004), which introduced the envy-cycle elimination algorithm, which yields an EF1 allocation. It was formally introduced by Budish (2011).

The 1.00008 inapproximability result was by Lee (2017). The 1.45 approximation algorithm was given by Barman et al. (2018). Strong NP-hardness of k-EQUAL-SUM-SUBSET is shown in Cieliebak et al. (2008).

The MILP which approximates the log of the utility function at each integer point was introduced by Caragiannis et al. (2019). At the time, Mosek did not support exponential cones, and so they did not compare the MILP approach to directly solving the discrete Eisenberg-Gale program. The results shown here are the first direct comparison of the two, to the best of my knowledge.

A-CEEI was introduced by Budish (2011), and an implementation of A-CEEI used at Wharton was given by Budish et al. (2016). The proof of PPAD completeness was by Othman et al. (2016).

Further reading.

As with fair division in the previous chapter, Brams and Taylor (1996) has coverage of discrete fair allocation problems as well. The paper introducing MNW as a methodology for fair allocation by Caragiannis et al. (2016) is well-written and a good research-level introduction to the topic. A really nice overview talk targeted at a technical audience is given by Ariel Procaccia here: `https://www.youtube.com/watch?v=7lUtS-l9ytI`.

CEEI for combinatorial utilities is too recent of a topic to have textbooks covering it. The original paper by Budish (2011) is a good starting point, and the followup paper by Budish et al. (2016) gives a lot of practical details.

14
Power Flows and Equilibrium Pricing

This chapter introduces a new topic: electricity markets, and their associated optimization problems. As we shall see, both economics and optimization play a key role in modern electricity grids.

For the first hundred years or so of the existence of the US power grid, it was managed by what are called *vertically integrated* utilities. These were companies that generated, sold, and transferred electricity directly to users. Typically, these would also be monopolies, meaning that they were the only possible supplier in a given region. In contrast, the late 1990s and early 2000s saw what's usually referred to as the *deregulation*[1] of the electric grid.

In the deregulated markets, the choice of who generates electricity, and at what price, is made using auction-based mechanisms where the auctioneer is an *independent system operator* (ISO). ISOs are quasi-governmental entities whose charter is to operate the grid, including deciding who generates what using auctions. The overarching market setup is very complicated. For example the New York market uses two electricity auctions: a spot auction every five minutes (which decides on the allocation of generation and purchasing for the next five minutes), and a day-ahead auction every hour (which allocates power generation and purchasing for that hourly interval of the following day), as well as several *capacity auctions* meant to ensure that the grid has sufficient generation capacity. We will focus more on the electricity auctions in this chapter. This chapter will start by introducing the operational optimization problems that ISOs need to solve on a continuous basis.

Compared to other markets, the electric grid has many peculiarities. For example:

(i) The grid operates in a continuous fashion, whereas the spot markets are

[1] This name is arguably misleading, as the electric grid is still highly regulated industry. Better terms would be restructuring, or decompositioning.

cleared in 5 minute intervals. This means that the outcome from a given auction is only approximately what will actually happen.

(ii) Supply (power generation) and demand (load generated by users) must be balanced at all times. The system will collapse if these quantities are not kept in check.

(iii) Goods (electricity) is generated at particular locations, and must be "transported" to the point of usage, potentially with a loss in power, or with congestion of the wires.

(iv) Electricity should be thought of as a "flow" in a network; therefore it's generally not possible to say that a particular user takes electricity from a particular plant. Both simply take electricity in and out of the "pool."

(v) Different types of electricity generators (e.g. wind, gas, nuclear) all have very different operating constraints, and thus differ in their ability to increase or decrease productions, and the speed at which they can do so.

These peculiarities are good to keep in mind when thinking about the grid and its markets, because they mean that e.g. incentives can be a tricky subject.

14.1 Optimal Power Flow

We now introduce the *optimal power flow* (OPF) problem. In OPF, we are given a directed network V, E of nodes and edges representing the electric grid in question. The set of nodes V in power parlance is called the set of *buses*. We will use *nodes* and *buses* interchangeably. The buses should be thought of as important locations in the physical grid, e.g. generation points, load points, or substations. The set of edges E is the connections between buses. In power parlance, these are called transmission *lines*. We will also use *edges* and *lines* interchangeably. We let E_i be the set of edges departing bus i.

Unlike every other section of this book, we will briefly need to work with complex numbers in this section. To that end, we let $\mathbf{i}$ refer to the imaginary unit satisfying $\mathbf{i}^2 = -1$. We will also use the notation z^* to refer to the complex conjugate of a complex number z.

These complex numbers arise because the power that flows through an electrical line is a complex number. The real part of this number is the *real power* that flows through the line, and the imaginary part is the *reactive power*. The real power is the "useful" part, it is the power that is eventually consumed by users, and it will be represented by the real part of the complex variable describing the power flow on a line. The reactive power is needed to maintain the voltage levels in the grid, and is not consumed by the end users. It arises from the alternating current flowing back and forth in a circuit. It will be represented by the imaginary part of the complex variable describing the power flow on a line.

The alternating current OPF (ACOPF) problem is a nonconvex quadratic optimization problem which models the physics of the power flow problem, including the fact that complex variables are needed. In particular, the net addition or removal of flow at a bus i will be a complex variable $p_i + \mathbf{i}q_i$, and similarly the power flow on a line $(i, j) \in E$ will be a complex variable $p_{ij} + \mathbf{i}q_{ij}$. We will mostly work with a linearization of this model, but I want to briefly describe it, so that you are aware of the approximation that is being made in the eventual LP we will use. To represent the problem, we will need the following variables:

- v_i is a complex number describing the voltage at bus $i \in V$.
- p_i is a real number describing the difference between generation and demand of *real* power at bus $i \in V$.
- q_i is the complex part of the difference between generation and demand of *reactive* power at bus $i \in V$.
- p_{ij} is the real part of the power flow on line $i, j \in E$; $p_{ij} > 0$ means power is flowing from i to j and $p_{ij} < 0$ means power flows the opposite direction.
- q_{ij} is the reactive power flow on line $i, j \in E$.

We will also need the following constants:

- $y_{ij} = g_{ij} + \mathbf{i}b_{ij}$ is a complex number describing the *admittance* of the line i to j.
- $\underline{v}_i, \overline{v}_i$ are lower and upper bounds on the voltage at bus i.
- Each bus $i \in V$ is subject to box constraints on its real power $\underline{p}_i, \overline{p}_i$, and reactive power $\underline{q}_i, \overline{q}_i$.
- Each line $i, j \in E$ is subject to a bound $\bar{s}_{ij}$ on the apparent power flow $p_{ij}^2 + q_{ij}^2$.

With all that, the ACOPF problem looks as follows, where f is some objective functions that we wish to optimize subject to the power flow constraints:

$$
\begin{aligned}
\min_{v,p,q} \quad & f(v,p,q) \\
\text{s.t.} \quad & p_{ij} + \mathbf{i} q_{ij} = v_i (v_i^* - v_j^*) y_{ij}^*, && \forall (i,j) \in E \\
& p_{ij}^2 + q_{ij}^2 \le \overline{s}_{ij}, && \forall (i,j) \in E \\
& \sum_{j \in E_i} p_{ij} = p_i, && \forall i \in V \\
& \sum_{j \in E_i} q_{ij} = q_i, && \forall i \in V \\
& p_i \in [\underline{p}_i, \overline{p}_i], q_i \in [\underline{q}_i, \overline{q}_i], |v_i| \in [\underline{v}_i, \overline{v}_i] && \forall i \in V
\end{aligned} \tag{ACOPF}
$$

The above problem is a very difficult optimization problem. In particular, even if f is a linear function, the first constraint is a nonconvex quadratic constraint, which makes the problem NP-hard in general. This leads to several problems, including the fact that this problem is typically too hard to solve to optimality for real-world OPF problems. It also means that we do not have strong duality, and strong duality is a critical component in the market-equilibrium-based mechanism used for pricing electricity.

Ideally, power system operators would solve Eq. (ACOPF) every five minutes in order to make dispatch decisions. However, because of the difficulty of solving ACOPF, they typically solve the linearized problem that we present in the next section. The ACOPF problem is sometimes solved for day-ahead markets, but even there they may resort to approximations. This leads to a variety of inefficiencies both in terms of operational efficiency, and in terms of market efficiency.

14.2 Linearized Power Flow

Going forward, we will work with a simplified model of power flows, which linearizes the nonconvex quadratic constraint in Eq. (ACOPF). We will call this model *DC power flow* (DCOPF), though this terminology is misleading, because it does not actually model direct-current power flows. Instead, it is simply a linearized approximation to AC power flows. There are many DCOPF models in the literature, what we cover is a representative one, but different modeling assumptions lead to different models.

A DCOPF model is obtained by making a number of simplifying assumptions of Eq. (ACOPF). First, because reactive power is negligible relative to real power, we set all reactive power variables to zero, meaning that we can remove all q variables and associated constraints.

Next, we write the complex variables using polar coordinates $v_i = m_i e^{\mathbf{i}\theta_i}$ for each i and apply Euler's formula (Eq. (A.4)). Then, we get the following equation for the real part of the nonconvex equation:

$$p_{ij} = g_{ij} m_i^2 - m_i m_j \left(g_{ij} \cos(\theta_i - \theta_j) - b_{ij} \sin(\theta_i - \theta_j) \right).$$

Then, we assume that all voltage magnitudes equal to one, i.e. $|m_i| = 1$. Finally, we set $g_{ij} = 0$ because $g_{ij} \ll b_{ij}$.

After making all these simplifications, the DCOPF problems has only linear constraints:

$$\begin{aligned}
\min_{\theta, p} \ & f(\theta, p) \\
\text{s.t. } & p_{ij} = b_{ij}(\theta_i - \theta_j), && \forall (i,j) \in E \\
& \sum_{j \in E_i} p_{ij} = p_i, && \forall i \in V \\
& p_i \in [\underline{p}_i, \overline{p}_i], && \forall i \in V \\
& |p_{ij}| \le \overline{s}_{ij}, && \forall (i,j) \in E.
\end{aligned} \tag{DCOPF}$$

If f is also a linear function, then Eq. (DCOPF) is an LP. In the formulation given here, each node $i \in V$ has a single power flow p_i into it (if $p_i > 0$) or out of it (if $p_i < 0$).

14.3 Economic Dispatch

In practice, nodes are often thought of as locations that potentially have both generators and demands. While Eq. (DCOPF) is completely general, it will be more convenient to explicitly include these multiple types of generators and demands in the model. Let Ψ_i^D be the set of demands at node i, where each demand $d \in \Psi_i^D$ has some utility u_d of receiving power, and some upper bound $\overline{p}_d$ on how much power they can consume. Similarly, let Ψ_i^G be the set of generators at node i, where each generator $g \in \Psi_i^G$ has some cost c_d of generating power, and a maximum generating capacity $\overline{p}_g$. We focus on a linear model of utility for simplicity; in practice nonlinear concave utilities for consumption and convex cost functions (e.g. quadratic cost functions for thermal generators) are sometimes used. The framework extends readily to this more general setting. If we now set our objective f to be equal to the social welfare of the resulting allocation, we get the following LP:

$$
\begin{aligned}
\max_{\theta,p} \ & \sum_{i\in V}\left(\sum_{d\in\Psi_i^D} u_d p_d - \sum_{g\in\Psi_i^G} c_g p_g\right) \\
\text{s.t. } & p_{ij} = b_{ij}(\theta_i - \theta_j), && \forall (i,j)\in E \\
& \sum_{j\in E_i} p_{ij} = \sum_{g\in\Psi_i^G} p_g - \sum_{d\in\Psi_i^D} p_d, && \forall i\in V \\
& p_d \in [0,\overline{p}_d], && \forall i\in V, d\in\Psi_i^D \\
& p_g \in [0,\overline{p}_g], && \forall i\in V, g\in\Psi_i^G \\
& |p_{ij}| \le \overline{s}_{ij}, && \forall (i,j)\in E.
\end{aligned}
\tag{14.1}
$$

A solution of this LP is referred to as *economic dispatch* because it maximizes social welfare over buyers and sellers. Let λ_i^* be the dual variable associated to the second equality in Eq. (14.1) in an optimal solution. Then λ_i^* can be thought of as the *locational marginal price* (LMP) of electricity at node i: each demand at i is charged this price, and each generator at i is paid this price per unit of electricity. In fact, a variant of this LP that takes into account additional operational constraints is used for pricing in many real-world electricity markets.

14.3.1 Market Equilibrium Interpretation

We now show that economic dispatch has a market equilibrium interpretation. We will use the Lagrange multiplier λ_i^* as the price of electricity at a given node $i \in V$. The allocation will be the one output by Eq. (14.1). Then we wish to show that under these prices, every market participant at a given node i optimizes their own utility given the local price λ_i^*. If we consider the Lagrangified problem using the optimal dual variables λ_i^*, we get the problem

$$
\begin{aligned}
\max_{\theta,p} \ & \sum_{i\in V}\left(\sum_{d\in\Psi_i^D} u_d p_d - \sum_{g\in\Psi_i^G} c_g p_g\right) + \sum_{i\in V}\lambda_i^*\left(\sum_{g\in\Psi_i^G} p_g - \sum_{d\in\Psi_i^D} p_d - \sum_{j\in E_i} p_{ij}\right) \\
\text{s.t. } & p_{ij} = b_{ij}(\theta_i - \theta_j), && \forall (i,j)\in E \\
& p_d \in [0,\overline{p}_d], && \forall i\in V, d\in\Psi_i^D \\
& p_g \in [0,\overline{p}_g], && \forall i\in V, g\in\Psi_i^G \\
& |p_{ij}| \le \overline{s}_{ij}, && \forall (i,j)\in E.
\end{aligned}
\tag{14.2}
$$

Now, if we consider the problem faced by an individual generator $g \in \Psi_i^G$

for some node i, in order to maximize their own utility they would like to solve the problem

$$\begin{aligned}\max_{p_g} \ & (\lambda_i^* - c_g)p_g \\ \text{s.t. } & p_g \in [0, \overline{p}_g].\end{aligned} \tag{14.3}$$

Comparing Eq. (14.2) and Eq. (14.3), we see that the variable p_g appears with the exact same set of constraints in both problems, and with the same coefficient in the objective. In other words, Eq. (14.2) decomposes along generators. Thus, by stationarity conditions we get that the value p_g^* from the economic dispatch solution is also optimal for the individual generator, if we set the price equal to λ_i^*. A completely analogous argument shows that each demand also maximizes its own utility.

It follows from the above that the prices and allocation from economic dispatch constitute a market equilibrium, in the sense that the supply of electricity equals the demand for electricity, and every market participant is receiving an allocation in their demand set given the prices.

14.3.2 Spatial Arbitrage

Finally, let us try to understand the transmission variables p_{ij}. To do so, we introduce the *spatial arbitrage problem*. Given the optimal dual variables λ^* this problem is defined as:

$$\begin{aligned}\max_{p_{ij},\theta} \ & \sum_{i \in V} \lambda_i^* \sum_{j \in E_i} p_{ij} \\ \text{s.t. } & p_{ij} = b_{ij}(\theta_i - \theta_j), \quad \forall (i,j) \in E \\ & |p_{ij}| \le \overline{s}_{ij}, \quad \forall (i,j) \in E.\end{aligned} \tag{14.4}$$

This can be thought of as a spatial arbitraging operation, where the variable p_{ij} is interpreted as the amount of electricity the arbitrageur purchases at node i in order to sell it at node j. Since $\sum_{j \in E_i} p_{ij} = \sum_{d \in \Psi_i^D} p_d - \sum_{d \in \Psi_i^G} p_g$, we know that $\lambda_i^* \sum_{j \in E_i} p_{ij}$ is the *excess* payment at node i, which can be either positive or negative. While individual line revenues for the arbitrageur may thus be positive or negative, Eq. (14.4) maximizes all the possible ways of transferring power across the network, given the prices. Since one possible feasible solution is to set all variables equal to zero, the spatial arbitrage revenue is nonnegative. By a similar decomposition argument as before, we see that the economic dispatch solution optimally solves the spatial arbitrage problem, since p_{ij} and θ appears with the same set of constraints and objective coefficients in both problems. Thus, if we let the transmission operator collect these excess payments, then the transmission operator acts as a spatial arbitrageur, who optimally tries to buy and sell power while satisfying the (linearized) transmission constraints. Since the arbitrage is nonnegative, the transmission operator is always budget balanced, and may collect additional payments. In practice, spatial arbitrage payments are used to fund grid operations such as transmission infrastructure and system reliability. They may also be used to compensate generators for providing capacity guarantees.

14.3.3 Economic Dispatch as a Mechanism

The economic dispatch framework derived in this section gives us a way to use markets to allocate power consumption and generation:

- Have every demand and generator submit their utility per unit of electricity, along with the consumption and generation caps.
- Compute an economic dispatch solution to decide which generators and demands get allocated.
- Charge everyone according to the dual prices.

This is how allocation and pricing is performed in many of the *spot markets* used by various ISOs. Spot markets run on a frequent basis (e.g. every five minutes), and determine generation and consumption for any *uncommitted* load and generation capacity. I stress the uncommitted part here, because some generators and demands will already have entered binding contracts on price and quantity in earlier markets, such as the day-ahead market.

We now investigate a few properties that would be nice to have for this market.

- **Truthfulness**: Unfortunately this mechanism is not truthful: while each participant acts optimally *given* the prices, they can themselves influence how those prices are set. This is already observed in a network with a single node; with a single node it is straightforward to see that some pair of generator and demander end up being the two entities setting the marginal price. That generator may then be incentivized to submit a slightly higher cost of generation in order to increase the price (and vice versa for the demander).
- **Efficiency**: If the submitted bids are truthful, then we would get efficiency since the economic dispatch model maximizes social welfare. That said, we already noted that this mechanism is easily seen to not be truthful. A second concern for efficiency is that we introduced a lot of approximations in order to arrive at an LP. It is not clear what those approximations do to the truthfulness or efficiency of the mechanism.

- **Budget balance**: The ISO needs to ensure that after paying generators and charging demands it ends up with a nonnegative amount of leftover money. However, we already saw in the spatial arbitrage section that the excess payments are captured via the p_{ij} variables, and the spatial arbitrager can make their utility at least zero, so revenue adequacy is guaranteed. ISOs are typically not allowed to make money either; for that reason the money made from spatial arbitrage is usually thought of as going to the providers of the transmission network, or towards additional investment in the network.

- **Individual rationality**: Is every participant better off participating in the market, as compared to simply exiting? It is easy to see from the market equilibrium conditions that every participant is incentivized to participate, as long as participants do not overstate their capacity, or report utilities/costs that are respectively higher/lower than their true values.

In addition to the approximations that we made going from ACOPF to DCOPF, this chapter also made some implicit assumptions. One of the biggest is that every generator can choose in continuous fashion how much electricity to produce. In practice, generators have various types of constraints on how they can change their output. For example, several types of energy producers require a long time to ramp production up or down (up to a day), and they may have minimum generation levels for when they are turned on. This is the case for several traditional generators such as nuclear and coal. Natural gas also has similar constraints. This introduces a discrete nature into the problem: we may need a day or more to reach certain production levels, and so the real-time market is operating "too late" for some decisions to be made. Renewables also have different types of constraints on their production, that depend on the type of renewable. For example, wind generators are not necessarily able to adjust their output at all, and are thus required to produce electricity at whatever level the weather dictates. This can even lead to negative energy prices, depending on whether we have a cost-free way of handling excess power.

All these constraints, as well as a general desire on the part of market participants for a certain amount of predictability in their revenues, necessitate additional market mechanisms that allow us to settle some generation and consumption further in advance than the spot market allows. This motivates the use of day-ahead markets, which we will study in Section 14.4.

14.4 Unit Commitment

So far, we have talked about the economic dispatch problem as if we solve it once, using a simple LP for finding the optimal generation and demand allocations. However, this is not how the ISOs actually decide on how to allocate. Instead, as mentioned briefly, there are several stages of allocation at various points in time. A key issue mentioned above is that many types of power-generating plants require long startup and shutdown times (on the order of hours to a day). This is one reason to consider day-ahead (DA) markets, where we commit some plants to producing energy on the following day, based on predicted demand. Beyond startup and shutdown times, another attractive property of DA markets is that they reduce uncertainty for the parties that settle on generation and load taking in the DA market. This may, for example, simplify staff scheduling.

14.4.1 Pricing via Linear Relaxation of Integer Variables

In this section we study how to handle binary operational decisions. For example, a nuclear or coal power plant must decide ahead of time whether to commit to turning the plant on or not. If they do commit, they usually have some minimum power output level (in addition to an upper bound), and if they do not, then they cannot generate any power. This binary decision problem obviously causes some problems for our market-based mechanism from Chapter 14: we used strong duality to get locational marginal prices for each node in the network. But with binary variables, we will not have strong duality! This section will discuss a few potential remedies to this problem, though none of them are perfect.

For simplicity, let us consider a single-node problem, where demand is fixed at p_d. We extend the generator model from earlier by letting each generator have some cost $C_g \geq 0$ of "switching on" their power generation. Switching on the plant will be represented by a binary variable $z_g \in \{0, 1\}$. If we write the economic dispatch problem with this new model, we get the following mixed-integer linear program (MILP):

$$
\begin{aligned}
\min_{p,z} \quad & \sum_{g \in \Psi^G} c_g p_g + C_g z_g \\
\text{s.t.} \quad & \sum_{g \in \Psi^G} p_g \geq p_d \\
& p_g \leq z_g \overline{p}_g, && \forall g \in \Psi^G \\
& p_g \geq z_g \underline{p}_g, && \forall g \in \Psi^G \\
& z_g \in \{0, 1\} && \forall g \in \Psi^G.
\end{aligned}
\tag{14.5}
$$

Suppose we solve this problem, and get a set of optimal binary variables z^*. Now we want to find a set of prices that support our efficient allocation. It turns out that we can in fact construct prices using these binary variables. The idea is to consider the continuous relaxation of Eq. (14.5), and then constrain each continuous variable z_g to take on exactly the value z_g^*. If we then apply strong duality to this new LP with an additional constraint, we still get a Lagrange multiplier on the demand constraint for setting prices, but we also get additional Lagrange multipliers acting as generator-specific payments or charges based on the newly-added constraint. Formally, we get the following LP:

$$
\begin{aligned}
\min_{p,z} \quad & \sum_{g \in \Psi^G} c_g p_g + C_g z_g \\
\text{s.t.} \quad & \sum_{g \in \Psi^G} p_g \geq p_d \\
& p_g \leq z_g \overline{p}_g, && \forall g \in \Psi^G \\
& p_g \geq z_g \underline{p}_g, && \forall g \in \Psi^G \\
& z_g = z_g^* && \forall g \in \Psi^G.
\end{aligned}
\tag{14.6}
$$

Now consider an optimal solution x^*, z^*, and let λ^* be the corresponding Lagrange multiplier on the first constraint in Eq. (14.6) and μ_g^* be the Lagrange multiplier for the last constraint in Eq. (14.6) for each g. We will set the payment for one unit of electricity at λ^*, and for each generator g such that $z_g^* = 1$, we pay them μ_g^* for turning on (or charge them $-\mu_g^*$ if μ_g^* is negative).

This turns out to yield a market equilibrium, as we will now show. Consider a generator g. They wish to solve the following problem:

$$\begin{aligned}\max_{p_g, z_g} \sum_{g \in \Psi^G} & (\lambda^* - c_g) p_g + (\mu_g - C_g) z_g \\ \text{s.t. } & p_g \le z_g \overline{p}_g \\ & p_g \ge z_g \underline{p}_g \\ & z_g \in \{0, 1\}.\end{aligned} \tag{14.7}$$

One way to solve this problem is to make z_g continuous, and hope that an integral solution happens to pop out. That yields the following program

$$\begin{aligned}\max_{p_g, z_g} \sum_{g \in \Psi^G} & (\lambda^* - c_g) p_g + (\mu_g - C_g) z_g \\ \text{s.t. } & p_g \le z_g \overline{p}_g \\ & p_g \ge z_g \underline{p}_g \\ & z_g \in \mathbb{R}.\end{aligned} \tag{14.8}$$

Clearly an optimal solution to this problem upper bounds the optimal solution to the integral version. But now it is easy to see that if we form the Lagrangian of Eq. (14.6):

$$\begin{aligned}&\min_{p,z} \sum_{g \in \Psi^G} c_g p_g + C_g z_g + \lambda^* \left(p_d - \sum_{g \in \Psi^G} p_g \right) + \sum_{g \in \Psi^G} \mu_g^* \left(z_g^* - z_g \right) \\ &\text{s.t.} \\ &\quad p_g \le z_g \overline{p}_g, \qquad \forall g \in \Psi^G \\ &\quad p_g \ge z_g \underline{p}_g, \qquad \forall g \in \Psi^G,\end{aligned} \tag{14.9}$$

then we get a problem which includes exactly the same constraints on p_g, z_g, and has the same coefficients in the objective. But then by strong duality we know that $p_g = p_g^*, z_g = z_g^*$ is an optimal solution to this problem, which shows that it must be an optimal solution to the LP for generator i. Since $z_g = z_g^*$, this LP solution satisfies the integrality condition. It follows that the generator receives an allocation in their demand set.

While the above approach was described in the context of unit commitment (i.e. a single Boolean "turning on" decision), it works much more broadly. If a generator has multiple binary decision then we can simply add one constraint per decision, and we will then get a price for each of their binary decisions.

One drawback of this pricing approach is that it tends to produce highly volatile prices, which can be both negative and positive. This can lead to prices that can seem very unfair (and materialize suddenly through minor changes to the pricing problem). A second concern is that we may no longer have budget balance, meaning that the ISO could potentially fall short on money due to the unit commitment prices. Third, a cost allocation issue arises, where it is not clear which consumers of electricity should be responsible for paying the start-up and shut-down costs.

14.4.2 Uplift Payments

In practice, ISOs often use what are called *uplift payments*. Uplift payments are an asymmetric variant of the previous pricing approach. The ISO will compute only locational marginal prices. Then, for generators with discrete decisions such as unit commitment, if the LMPs do not support their assigned decisions and power output, the ISO will pay the difference. Note that this can make the generator better or worse off depending on context. For example, μ_g being negative is ignored which helps the generator, but when μ_g is positive the uplift payment could be smaller than μ_g still.

14.4.3 Convex Hull Pricing

An alternative pricing approach is that of *convex hull pricing* (CH pricing). CH pricing is very easy to set up. We simply Lagrangify the demand constraint, and solve the resulting minimization problem over electricity prices. Formally, we solve

$$\min_{\lambda} q(\lambda),$$

where $q(\lambda)$ is defined as

$$q(\lambda) := \begin{array}{lll} \min\limits_{p,z} & \sum\limits_{g \in \Psi^G} c_g p_g + C_g z_g + \lambda\big(p_d - \sum\limits_{g \in \Psi^G} p_g\big) & \\ \text{s.t.} & p_g \le z_g \overline{p}_g, & \forall g \in \Psi^G \\ & p_g \ge z_g \underline{p}_g, & \forall g \in \Psi^G \\ & z_g \in \{0,1\} & \forall g \in \Psi^G. \end{array}$$

From an optimization perspective this approach has some attractive properties, especially the fact that given a fixed λ, solving $q(\lambda)$ decomposes into simple per-generator optimization problems. On the other hand, since we do not have strong duality, this approach does not necessarily give us a feasible solution. In practice, the resulting CH prices λ^* would be extracted, but the allocation would use the original MILP for finding a feasible allocation. This means that in general CH pricing will not be such that generators get allocations that are in their demand set. To fix this issue, ISOs would then provide additional uplift payments.

14.4.4 Connecting DA and RT Markets

So far we have discussed RT and DA markets in isolation. In practice, the RT market operates after a number of contracts for consumption and generation have been settled in the DA market. For example, suppose a generator was assigned 100 megawatt (MW) of generation for an RT period, but it turns out that they will only be able to produce 97MW. In that case, the remaining 3MW must be purchased in the RT market. Financially speaking, the generator would then be viewed as having purchased 3MW of power in that RT market. Similarly, a demand that purchased 100MW of power in the DA market but then consumed only 90MW would be viewed as selling 10MW of power in the RT market. In general, we can view the RT market as a balancing operation that corrects any imbalances that occur due to increased or decreased consumption or generation specified in the DA market.

If not for uncertainty, it is easy to convince yourself that the price in the DA and RT markets should be the same. If they were not, then any generator that was assigned to generate in the market with the lower price would simply wish to change their bids such that they end up getting assigned the same generation in the market with the higher price. A similar argument holds for demands.

A key reason why the RT market may nonetheless require balancing is that consumer electricity usage as forecasted in the DA market will differ from the realized usage in the RT market. This causes relatively manageable imbalances in the market, and the ISO needs to correct these imbalances in order to keep the system functioning. A second and more severe imbalance issue that can occur is generator outages. A generator outage can lead to large imbalances that require significant additional generation allocation in the RT market.

Due to these imbalances, and the very short-term nature of the RT market, flexible generation and consumption entities will be rewarded at a higher rate in the RT market when realized demands turns out to be higher than realized generation. On the other hand, expensive generators that are primarily used to cover the case of excess demand in the RT market will not make any money when realized demand is lower than realized generation. Thus, the cost of generation for such plants is often high, which can lead to higher volatility in RT market prices.

14.5 Historical Notes

Schweppe et al. (1988) introduced the idea of spot pricing in the context of competitive markets for electricity and is credited with laying the theoretical foundations for electricity markets. Hogan (1992) introduced many of the ideas leading to the idea of pricing for transmission rights. The approach for pricing binary decision by using the MIP solution as constraints in the LP was introduced by O'Neill et al. (2005). Convex hull pricing was introduced by Gribik et al. (2007).

Further Reading

For the DCOPF problem, Stott et al. (2009) gives a comprehensive overview of different modeling assumptions, the resulting models, and their pros and cons. Wood et al. (2013) gives textbook coverage of these topics.

Taylor (2015) covers many of the optimization aspects of the power grid. This book also has some coverage of energy markets. Kirschen and Strbac (2018) has extensive coverage of the economic aspects of energy systems.

Sweeney (2013) provides a detailed account of the California energy crisis, which is an interesting story that highlights a number of bad market design decisions, many of them politically motivated. That crisis lead to severe blackouts, huge budget deficits for several energy companies (with one going bankrupt), and had large ramifications for the state budget.

Jalal Kazempour from the Danish Technical University has a set of slides and lecture videos[2] that give a nice optimization-based introduction to energy markets.

[2] Found here: `https://www.jalalkazempour.com/teaching`

PART FOUR

AUCTIONS AND INTERNET ADVERTISING MARKETS

15

Internet Advertising Auctions: Position Auctions

In this chapter we begin our study of another class of more advanced auction mechanisms, motivated by internet advertising auctions. Internet advertising auctions provide the funding for almost every free internet service such as Google Search, Facebook, Twitter, and so on. At the heart of these monetization schemes is a market design based around independently running auctions every time a user shows up. This happens many times per second, advertisers participate in millions of auctions, have budget constraints that span the auctions, and each user query generates multiple potential slots for showing ads. For all these reasons, these markets turn out to require a lot of new theory for understanding them. Moreover, the scale and speed of the problem necessitates the design of algorithmic agents for bidding on behalf of advertisers.

First we will introduce the *position auction*, which is a highly structured multi-item auction. It is designed to capture the fact that ads are usually shown as a ranked list as part of e.g. a Google query. We will look at the two most practically-important auction formats: the generalized second-price auction (GSP), and the Vickrey-Clarke-Groves (VCG) auction. Then in the following chapters, we will study auctions with budgets, and repeated auctions over time.

15.0.1 Considerations for internet advertising

Consider the following problem: a user shows up and searches for the word “mortgage” on Google; now, you are Google, and you have thousands of ads that you could show to the user when returning the search result. Typically, Google shows a few ads at the top of the search results (say 2 ads, to be concrete); an example is shown in Fig. 15.1. This setting is referred to as the “sponsored search setting.” How do you decide which ads to show? And how do you decide how much to charge each advertiser for showing their ad? A natural suggestion would be to try to use auctions. Based on earlier chapters of the

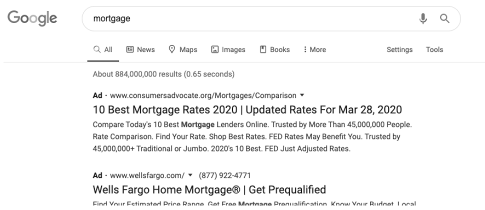

Figure 15.1 A Google query for "mortgage" shows 2 ads. Organic search results follow further down.

book, one might think of running four separate first or second-price auctions, one for each ad slot. In that case, it is clear how to decide winners and how to set prices. Yet this immediately runs into a problem: the same ad may win multiple auctions, and thus be shown in several slots. This looks bad for the user, the advertiser almost surely does not want to pay for multiple slots, and it is inefficient (in the sense that multiple advertisers could have generated value, rather than just the one). Instead, we need to design an auction format that allows multiple items to be allocated simultaneously. But we cannot simply use the multi-item generalization of e.g. the second-price auction, where each item is identical. This is because different slots are *not* identical: users are generally more likely to click on the first ad than the second ad, and so on. This motivates the *position auction*, which we study in this chapter.

The position auction model can also be used to approximate other settings such as the insertion of ads in a *news feed*; a news feed is the familiar infinitely-scrolling list of e.g. Facebook posts, Reddit posts, Instagram posts, or Twitter posts. For example, Reddit typically inserts one ad in the set of visible results before scrolling, with another ad appearing in the next 10-15 results (I tested this on June 24th 2025). Similarly, Facebook and Twitter insert 1-2 sponsored posts near the top of the feed. Truly capturing feed auctions does require some care, however. The assumption of there being a fixed number of items is incorrect for that setting. Instead, the number of ads shown depends on how far the user scrolls, the size of the ads, and what else is being shown in terms of organic content. We will focus on the simpler setting with a fixed number of slots; properly handling feed auctions is an interesting extension of what we discuss.

Beyond the multi-item, budget, and time aspects, internet advertising has a few other interesting quirks. These are discussed briefly below, though we will mostly abstract away considerations around these issues.

Targeted advertising.

In a classical advertising setting such as TV or newspaper advertising, the same ad is shown to every viewer of a given TV channel, or every reader of a newspaper. This means that it is largely not feasible for smaller, and especially niche, retailers to advertise, since their return on investment is very low due to the small fraction of viewers or readers that fit their niche. All this changed with the advent of internet advertising, where niche retailers can perform much more fine-grained targeting of their ads. This has enabled many niche retailers to scale up their audience reach significantly.

One way that targeting can occur is directly through association with the search term in sponsored search. For example, by bidding on the search term "mortgage," a lender is effectively performing a type of targeting. However, a second type of targeting occurs by matching on query and user features (such targeting is used across many types of internet advertising including search, feed ads, and others). For example, a company selling surf boards might wish to target users at the intersection of the categories {age 16-30, lives in California}. Because each individual auction corresponds to a single user query, the idea of targeted advertising can be captured in the valuations that we will use for the buyers in our auction setup: each buyer corresponds to an advertiser, each auction corresponds to a query, and the buyer will have value zero for all items in a given auction if the associated query features do not match their targeting criteria.

Targeted advertising has the potential for some adverse effects. Of particular note are demographic biases in the types of ads being shown (a well-documented example is that in some settings, ads for new luxury housing developments were disproportionately shown to certain demographics). In Chapter 18 we will study some questions around demographic fairness.

Pay per click.

Another revolution compared to pre-internet advertising is the *pay per click* nature of most internet advertising auctions. Many advertisers are not actually interested in the user simply viewing their ad. Instead, their goal is to get the user to click on the ad, or even something downstream of clicking on the ad, such as selling the advertised product via the linked website. Because the platform, such as Google, is in a much better position to predict whether a given user will click on a given ad, these auctions operate on a *cost per click* basis, rather than a *cost per impression*.[1] What this means is that any given advertiser does not

[1] An *impression* is industry lingo for the user being shown the ad, regardless of whether they interact with it.

actually pay just because they won the auction and got their ad shown, instead they pay *only* if the user actually clicks on their ad.

From an auction perspective, this means that the valuations used in the auctions must take into account the probability that the user will click on the ad. Valuations are typically constructed by breaking down the value that a buyer i (in this case an advertiser) has for an item (which is a particular slot in the search query or user feed) into several components. The *value per click* of advertiser i is the value $v_i > 0$ they place on any user within their targeting criteria clicking on their ad (modern platforms generalize this concept to a value per *conversion*, where a conversion can be a click, an actual sale of a product, the user viewing a video, etc.). The *click-through-rate* is the likelihood that the user behind query j will click on the ad of advertiser i, independently of where on the page the ad is shown. We denote this by CTR_{ij}; we will assume that $\text{CTR}_{ij} = 0$ if query j does not fall under the targeting criteria of buyer i. Finally, the *slot qualities* $q_1, \ldots, q_S$ are scalar values denoting the quality of each slot that an ad could end up in. These are monotonically decreasing values, indicating the fact that it's generally preferable to be shown higher up on the page. Now, finally, the value that buyer i has for being shown in slot s of query j is modeled as $v_{ijs} = v_i \cdot \text{CTR}_{ij} \cdot q_s$.

For the rest of the chapter, we will assume that we can work directly with a value $v_{ij} = v_i \cdot \text{CTR}_{ij}$ which captures the value that buyer i has for auction j; this value encodes the value per click, the CTR, and the targeting criteria (but can allow for more general valuations that do not decompose). Then, the buyer's valuation for a particular slot can be written as $v_{ijs} = v_{ij} \cdot q_s$. Working directly with v_{ij} implicitly assumes correct CTR predictions, which is obviously not true in practice. In practice the CTRs are estimated using machine learning, and it is of interest to understand which discrepancies this introduces into the market. Secondly, we are assuming that buyers are maximizing their expected utility, rather than observed utility. This is largely a non-problem; because advertiser participate in thousands or even millions of auctions, the law of large numbers implies that their realized value can reasonably be expected to match the expectation (at least if the CTR predictions are correct). The slot quality q_s will be handled separately in the next section.

15.1 Position Auctions

In the position auction model, a set of S *slots* on a fixed item j are for sale. Because we are analyzing this individual auction in isolation, we can drop the j index and simply assume that v_i gives the expected value per click for buyer

i in the current auction. The slots in the auction are shown in ranked order, and the value that an advertiser derives from showing their ad in a particular slot s decomposes into two terms $v_{is} = v_i q_s$ where v_i is the value that the advertiser places on a user clicking on their ad, and q_s is the advertiser-independent click probability of slot s. Here we assume that v_i already incorporates the click-through rate (so in particular it could be that $v_i = v'_i \cdot \text{CTR}_i$ where v'_i is their actual value per click, and CTR_i is the click-through rate in the current auction). It is assumed that $q_1 \geq q_2 \geq \cdot \geq q_S$, i.e. the top slot is better than the second slot, and so on. Going back to the original setting, a position auction corresponds to the individual auction that is run when a particular user query shows up.

Now suppose that the n advertisers submit bids $b \in \mathbb{R}^n_{\geq 0}$. Both auction formats we will use then proceed to perform allocation via welfare maximization, assuming that the bids are truthful. We will also refer to this as bid maximization. In order to maximize (reported) welfare, we sort the bids b (suppose without loss of generality that $b_1 \geq b_2 \geq \cdots \geq b_n$), and allocate the slots in order of bids (so buyer 1 with bid b_1 gets slot 1, buyer 2 gets slots 2, and so on up to bid b_S getting slot S).

Example 15.1 Suppose we have two slots with quality scores $q_1 = 1, q_2 = 0.5$, and three buyers with values $v_1 = 10,\ v_2 = 8,\ v_3 = 2$, and suppose they all bid their values. Then buyer 1 is allocated slot 1, and they obtain a value of $v_1 \cdot q_1 = 10$. Buyer 2 is allocated slot 2, and they generate a value $v_2 \cdot q_2 = 4$. Buyer 3 gets nothing.

15.1.1 Generalized Second-Price Auctions

The *generalized second-price* (GSP) auction sells the S slots as follows: First, we allocate via bid maximization as described above. If the user clicks on ad $i \leq S$, then advertiser i is charged the next-highest bid b_{i+1}. GSP generalizes second-price auctions in the sense that if $S = 1$ then this auction format is equivalent to the standard second-price auction (if we take expected values in lieu of the pay-per-click model). However, this is a fairly superficial generalization, since GSP turns out to lose the key property of the second-price auction: truthfulness!

In particular, consider Example 15.1 again, and suppose that buyers 2 and 3 bid truthfully. With GSP prices, buyer 1 pays $v_2 \cdot q_1 = 8$ and gets utility $q_1(v_1 - v_2) = 2$ when everyone bids truthfully. If buyer 1 instead bids some value between 2 and 8, then they get utility $q_2(v_1 - v_3) = 4$. Thus, buyer 1 is better off misreporting. More generally, it turns out that the GSP auction can have several pure-Nash equilibria, and some of these lead to allocations that are not welfare-maximizing. Consider the following bid vector for Example 15.1, $b = (4, 8, 2)$. Buyer 1 gets utility $0.5(10 - 2) = 4$ (whereas they'd get utility 2 for bidding above 8). Buyer 2 gets utility $1(8 - 4) = 4$ (whereas they'd get utility $0.5(8 - 2) = 3$ for bidding below 4). Buyer 3 is priced out.

In spite of the above, the GSP rule has been in widespread use in the internet advertising industry since 2002. See the historical notes for pointers to some interesting retrospectives on how GSP arose.

15.1.2 VCG for Position Auctions

The second pricing rule we will consider is the VCG rule. Recall from Chapter 3 that VCG computes the welfare-maximizing allocation (assuming truthful bids), and then charges buyer i their externality (i.e. how much the presence of buyer i decreases the social welfare across the remaining agents).

Let $W^S_{-i} = \sum_{k=1}^{i-1} b_k q_k + \sum_{k=i+1}^{S+1} b_k q_{k-1}$ be the social welfare achieved by buyers $[n] \setminus \{i\}$ if we maximize welfare across only those buyers, and let $W^{S-i}_{-i} = \sum_{k=1}^{i-1} b_k q_k + \sum_{k=i+1}^{S} b_k q_k$ be the social welfare of $[n] \setminus \{i\}$ if we maximize welfare using all slots except slot i. Buyer i gets charged their externality, which is as follows:

$$W^S_{-i} - W^{S-i}_{-i} = \sum_{k \in \{i+1,\dots,S+1\}} b_k \cdot q_{k-1} - \sum_{k \in \{i+1,\dots,S\}} b_k \cdot q_k \tag{15.1}$$

$$= \sum_{k \in \{i+1,\dots,S+1\}} b_k \cdot (q_{k-1} - q_k). \tag{15.2}$$

Where we let $q_{S+1} = 0$. We already saw a sketch of the fact that VCG is incentive compatible in Chapter 3, but here we show the result specifically for the position auction setting, where the proof is nice and short.

Theorem 15.2 *The VCG auction for position auctions is incentive compatible.*

Proof Suppose again that buyer bids are sorted, with buyer i winning slot i when bidding truthfully. Now suppose buyer i misreports and gets slot k instead. Now we want to show that bidding truthfully maximizes utility, which means:

$$q_i \cdot v_i - [W^S_{-i} - W^{S-i}_{-i}] \geq q_k \cdot v_i - [W^S_{-i} - W^{S-k}_{-i}].$$

Simplifying this expression gives

$$q_i \cdot v_i + W^{S-i}_{-i} \geq q_k \cdot v_i + W^{S-k}_{-i}.$$

Now we see that both the right-hand and left-hand sides correspond to social welfare under two different allocations (where we treat bids from other buyers as their true value). The left-hand side is social welfare when i bids truthfully, while the right-hand side is social welfare when i misreports in a way that gives them slot k. Given that VCG picked the left-hand side, and VCG allocates via welfare maximization, the left-hand side must be larger. □

15.2 Historical Notes

An early version of the GSP auction was introduced in the early internet search days at Overture, which was an innovator in sponsored search advertising, and they were later acquired by Yahoo, which used this rule as well. Google then started using the more modern version of GSP. From an academic perspective, the GSP rule and position auctions in general started to be studied by Varian (2007) and Edelman et al. (2007), motivated by its use in practice. An interesting historical perspective on why VCG was not chosen is discussed by Varian and Harris (2014) who worked at Google at the time. The primary reasons are essentially inertial: a lot of engineering work was already going into GSP, and advertisers had gotten used to bidding in GSP. A major concern would be that they would need to raise their bids in VCG due to its truthfulness, which might be hard to explain to them given their existing experience with GSP. Facebook notably uses VCG rather than GSP (Varian and Harris, 2014), unlike the prior internet companies.

Further reading.

Easley et al. (2010) and Nisan et al. (2007) both have a few chapters on internet advertising auctions. Devanur and Mehta (2023) surveys a number of topics that we did not cover here, including the AdWords problem and a famous approximation algorithm for that problem.

16

Auctions with Budgets and Pacing Equilibria

The previous chapter introduced a new aspect to auctions associated with internet advertising auctions: the multiple-slot issue. This chapter studies a second major practical aspect of internet advertising auctions: budgets. In these auctions, most advertisers specify a budget constraint that must hold in aggregate across all the payments made by the advertiser over a given period of time. Because these budget constraints are applied across all the auctions that an advertiser participates in, they couple the auctions together, and force us to consider the aggregate incentives across auctions. This is in contrast to all of our previous auction results, which studied a single auction in isolation. Notably, these budgets constraints break the incentive compatibility of the second-price auction; for an advertiser with a budget constraint, it is not necessarily optimal to bid their true value in each auction!

16.1 Auction Markets

Throughout the rest of this chapter, we will consider settings where each individual auction is a single-item auction, using either first or second-price rules. This is of course a simplification: in practice each individual auction would be more complicated (e.g. a position auction), but even just for single-item individual auctions it turns out that there are a lot of interesting problems.

In this setting we have n buyers and m goods. Buyer i has value v_{ij} for good j, and each buyer has some budget B_i. Each good j will be sold via sealed-bid auction, using either first or second-price. We assume that for all buyers i, there exists some item j such that $v_{ij} > 0$, and similarly for all j there exists i such that $v_{ij} > 0$. Let $x \in \mathbb{R}^{n \times m}$ be an allocation of items to buyers, with associated prices $p \in \mathbb{R}^m$. The utility that a buyer i derives from this allocation is

$$u_i(x_i, p) = \begin{cases} \langle v_i, x_i \rangle - \langle p, x_i \rangle & \text{if } \langle p, x_i \rangle \leq B_i \\ -\infty & \text{otherwise} \end{cases}.$$

We call this setting an *auction market*. If second-price auctions are used then we call it a second-price auction market, and conversely we call it a first-price auction market if first-price auctions are used.

16.2 Second-Price Auction Markets

In Chapter 3 we saw that the second-price auction is incentive compatible. However, this relied on there being a single auction, and no budgets. It's easy to construct an example showing that this is no longer true in second-price auction markets. Consider a market with two buyers and two items, with valuations $v_1 = (100, 100), v_2 = (1, 1)$ and budgets $B_1 = B_2 = 1$. If both buyers submit their true valuations then buyer 1 wins both items, pays 2, and gets $-\infty$ utility.

Instead, each buyer needs to somehow smooth out their spending across auctions. For large-scale internet auctions this is typically achieved via some sort of *pacing rule*. Typically, these pacing rules are implemented by an *auto-bidder*, which is an algorithm offered by the platform, which bids on behalf the advertiser. The two most prevalent pacing rules are:

(i) *Probabilistic pacing*: each buyer i is given a parameter $\alpha_i \in [0, 1]$ denoting the probability that they should participate in each auction. For each auction j, an independent coin is flipped which comes up heads with probability α_i, and if it comes up heads then the buyer submits a bid $b_{ij} = v_{ij}$ to that auction. If it comes up tails then they do not bid in the auction.
(ii) *Multiplicative pacing*: each buyer i is given a parameter $\alpha_i \in [0, 1]$, which acts as a scalar multiplier on their truthful bids. For each auction j, buyer i submits a bid $b_{ij} = \alpha_i v_{ij}$.

Both are offered by the major internet advertising platforms such as Google and Meta. Figure 16.1 shows a comparison of pacing methods for a simplified setting where time is taken into account. Here we assume that we are considering some buyer i whose value is the same for every item, but other bidders are causing the items to have different prices. On the x-axis we plot time, and on the y-axis we plot the price of each item. On the left is the outcome from naive bidding: the buyer spends their budget too fast, and ends up running out of budget when there are many high-value items left for them to buy. In practice,

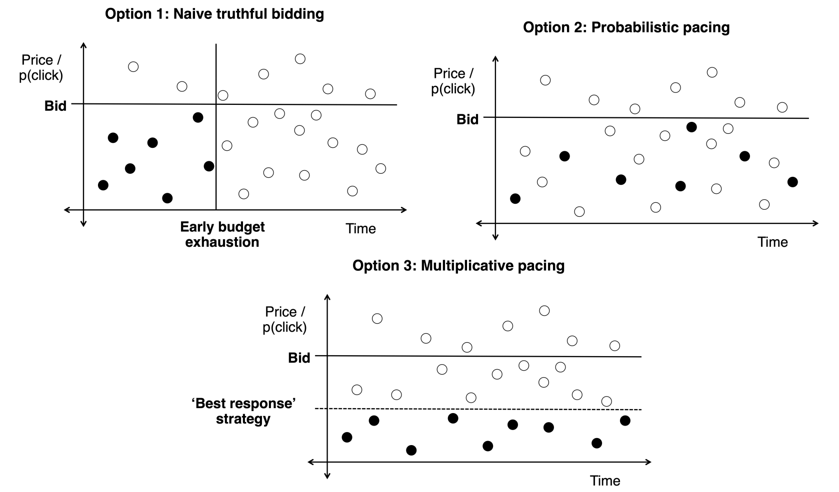

Figure 16.1 Comparison of pacing methods. The x-axis displays time. The y-axis is the price divided by the click probability for each opportunity. Each circle denotes an impression. Hollow circles represent impression not won, while filled-in circles denote impressions that are won. The horizontal bid line represents the truthful bid without any modifications. Left: no pacing, right: probabilistic pacing, bottom: multiplicative pacing.

many buyers also prefer to smoothly spend their budget throughout the day. In the middle we show probabilistic pacing, where we do get smooth budget expenditure. However, the buyer ends up buying some very expensive item, while missing out on much cheaper items that have the same value to them. Finally, on the right is the result from multiplicative pacing, where the buyer picks an optimal threshold to buy at, and thus buys item optimally in order of bang-per-buck. In this chapter we will focus on multiplicative pacing, but see the historical notes section for some references to papers that also consider probabilistic pacing.

The intuition given in Figure 16.1 can be shown to hold more generally. Given a set of bids by all the other bidders, a buyer can always implement a best response by choosing an optimal pacing multiplier:

Proposition 16.1 *Suppose we allow arbitrary bids in each auction. If we hold all bids for buyers $k \neq i$ fixed, then buyer i has a best response that consists of choosing a pacing multiplier (assuming that if a buyer is tied for winning an auction, they can specify the fraction that they win).*

Proof Since every other bid is held fixed, we can think of each item as having some price $p_j = \max_{k \neq i} b_{kj}$, which is what i would pay if they bid $b_{ij} \geq b_{kj}$. Now we may sort the items in decreasing order of bang-per-buck $\frac{v_{ij}}{p_j}$. An optimal allocation for i clearly consists of buying items in this order, until they reach some index j such that if they buy every item with index $l < j$ and some fraction x_{ij} of item j, they either spend their whole budget, or j is the first item with $\frac{v_{ij}}{p_j} \geq 1$ (if $\frac{v_{ij}}{p_j} > 1$ then $x_{ij} = 0$). Now set $\alpha_i = \frac{p_j}{v_{ij}}$. With this bid, i gets exactly this optimal allocation: for all items $l \leq j$ (which are the items in the optimal allocation), we have $\alpha_i v_{il} = \frac{p_j}{v_{ij}} v_{il} \geq \frac{p_l}{v_{il}} v_{il} = p_l$. □

The goal will be to find a *pacing equilibrium*:

Definition 16.2 A second-price pacing equilibrium (SPPE) is a vector of pacing multipliers $\alpha \in [0,1]^n$, a fractional allocation x_{ij}, and a price vector such that for every buyer i:

- For all j, $\sum_{i \in [n]} x_{ij} = 1$, and if $x_{ij} > 0$ then i is tied for highest bid on item j.
- If $x_{ij} > 0$ then $p_j = \max_{k \neq i} \alpha_k v_{kj}$.
- For all i, $\sum_{j \in [m]} p_j x_{ij} \leq B_i$. Additionally, if the inequality is strict then $\alpha_i = 1$.

The first and second conditions of pacing equilibrium simply enforce that the item always goes to winning bids at the second-price rule. The third condition ensures that a buyer is only paced if their budget constraint is binding. It follows (almost) immediately from Proposition 16.1 that every buyer is best responding in SPPE.

A nice property of SPPE is that it is always guaranteed to exist. This fact is not immediate from the existence of Nash equilibrium in convex games (see Chapter 10), since an SPPE corresponds to a specific type of pure-strategy Nash equilibrium:

Theorem 16.3 *An SPPE of a pacing game is always guaranteed to exist.*

We won't cover the whole proof here, but we will state the main ingredients, which are useful to know more generally.

- First, a smoothed pacing game is constructed. In the smoothed game, the allocation is smoothed out among all bids that are within ϵ of the maximum bid, thus making the allocation a deterministic function of the pacing multipliers α. Several other smooth approximations are also introduced to deal with other discontinuities. In the end, a game is obtained, where each player simply has as their action space the interval $[0,1]$ and utilities are nice continuous and quasi-concave functions.

Problem instance:

i	v_{i1}	v_{i2}	v_{i3}	v_{i4}	B_i
1	100	1	99	100	1
2	1	100	99		1
3				100	100

Equilibrium 1: **Revenue = 102**

α_i	b_{i1}	b_{i2}	b_{i3}	b_{i4}	spend
1	100	1	99	100	1
0.01	0.01	1	0.99		1
1				100	100

Equilibrium 2: **Revenue = 3**

α_i	b_{i1}	b_{i2}	b_{i3}	b_{i4}	spend
0.01	1	0.01	0.99	1	1
1	1	100	99		1
1				100	1

Figure 16.2 Multiplicity of SPPE. On the left is shown a problem instance, and on the right is shown two possible second-price pacing equilibria.

- Secondly, the fixed-point theorem for pure-strategy equilibrium existence in convex games is invoked (see Theorem 10.4). This guarantees existence of a pure-strategy Nash equilibrium in the smoothed game.
- Finally, the limit point of smoothed games as the smoothing factor ϵ tends to zero is shown to yield an equilibrium in the original pacing problem.

Unfortunately, while SPPE is guaranteed to exist, it turns out that sometimes there are several SPPE, and they can have large differences in revenue, social welfare, and so on. An example is shown in Figure 16.2. In practice this means that we might need to worry about whether we are in a "good" equilibrium (e.g. in terms of revenue or social welfare).

Another positive property of SPPE is that every SPPE is also a market equilibrium, if we consider a market equilibrium setting where each buyer has a quasi-linear demand function that respects the total supply as follows:

$$D_i(p) = \arg\max_{0 \le x_i \le 1} \langle v_i - p, x_i \rangle \text{ s.t. } \langle p, x_i \rangle \le B_i.$$

This follows immediately by simply using the allocation x and prices p from the SPPE as a market equilibrium. Proposition 16.1 tells us that $x_i \in D_i(p)$, and the market clears by definition of SPPE. This means that SPPE has a number of nice properties such as no (budget-adjusted) envy. Pareto optimality does not follow, because the buyers have satiating preferences, thus violating the condition for the first theorem of welfare economics (Theorem 11.3). Nonetheless, one can show that Pareto optimality is recovered if the seller's utility is included, and they are assumed to have linear utility in revenue, with no value for the items in the auctions.

Finally, we turn to the question of computing an SPPE. Unfortunately the news there is bad. It is known that computing an SPPE is a PPAD-complete problem, and thus in the same equivalence class of problems as the problem of computing a Nash equilibrium in a two-player general-sum game. Moreover, it is also known that we cannot hope for iterative methods to efficiently compute an approximate SPPE. Beyond merely computing *any* SPPE, we could also try to find one that maximizes revenue or social welfare. This problem turns out to be an NP-complete problem.

There is a mixed-integer program for computing SPPE (see Conitzer et al. (2022a)), but unfortunately it is not very scalable. In Conitzer et al. (2022a) the program does not scale beyond about 18 buyers and 18 goods.

In spite of the above issues, major internet platforms such as Google and Meta run markets whose autobidders attempt to implement a generalization of the second-price auction market, and it appears to work well in practice. It is unknown whether there is a nice model to explain why these issues are not a problem in practice.

16.3 First-Price Auction Markets

Next we consider what happens if we instead sell each item by first-price auction as part of an auction market.

First we start by defining what we call *budget-feasible pacing multipliers*. Intuitive, this is simply a set of pacing multipliers such that everything is allocated according to first-price auction, and everybody is within budget.

Definition 16.4 A set of *budget-feasible pacing multipliers* (BFPM) is a vector of pacing multipliers $\alpha \in [0,1]^n$ and a fractional allocation x_{ij} such that for every buyer i:

- Prices are defined to be $p_j = \max_k \alpha_k v_{kj}$.
- For all j, $\sum_{i \in [n]} x_{ij} = 1$, and if $x_{ij} > 0$ then i is tied for highest bid on item j.
- For all i, $\sum_{j \in [m]} p_j x_{ij} \leq B_i$.

Again, the goal will be to find a *pacing equilibrium*. This is simply a BFPM that satisfied the complementarity condition on the budget constraint and pacing multiplier.

Definition 16.5 A *first-price pacing equilibrium* (FPPE) is a BFPM (α, x) such that for every buyer i:

- For all i, if $\sum_{j \in [m]} p_j x_{ij} < B_i$ then $\alpha_i = 1$.

Notably, the only difference to SPPE is the pricing condition, which now uses first price. This seemingly-small change leads to improvement on all the issues we saw above, though at the cost of losing the pure-strategy Nash equilibrium property.

A very nice property of the first-price setting is that BFPMs satisfy a monotonicity condition: if (α', x') and (α'', x'') are both BFPM, then the pacing vector $\alpha = \max(\alpha', \alpha'')$ (where the max is taken component-wise) is also a BFPM. The associated allocation is that for each item j, we first identify whether the highest bid comes from α' or α'', and use the corresponding allocation of j (breaking ties towards α').

Intuitively, the reason that (α, x) is also BFPM is that for every buyer i, their bids are the same as in one of the two previous BFPMs (say (α', x') without loss of generality), and so the prices they pay are the same as in (α', x'). Furthermore, since every other buyer is bidding at least as much as in (α', x'), they win weakly less of each item (using the tie-breaking scheme described above). Since (α', x') satisfied budgets, (α, x) must also satisfy budgets. The remaining conditions are easily checked.

In addition to component-wise maximality, there is also a *maximal* BFPM (α, x) (there could be multiple x compatible with α) such that $\alpha \geq \alpha'$ for all α' that are part of any BFPM. Consider $\alpha_i^* = \sup\{\alpha_i | \alpha \text{ is part of a BFPM}\}$. For any ϵ and i, we know that there must exist a BFPM such that $\alpha_i > \alpha_i^* - \epsilon$. For a fixed ϵ we can take component-wise maxima to conclude that there exists $(\alpha^\epsilon, x^\epsilon)$ that is a BFPM. This yields a sequence $\{(\alpha^\epsilon, x^\epsilon)\}$ as $\epsilon \to 0$. Since the space of both α and x is compact, the sequence has a limit point (α^*, x^*). By continuity (α^*, x^*) is a BFPM.

We can use this maximality to show existence and uniqueness (of multipliers, not allocations) of FPPE:

Theorem 16.6 *An FPPE always exists and the set of pacing multipliers $\{\alpha\}$ that are part of an FPPE is a singleton.*

Proof Consider the component-wise maximal α and an associated allocation x such that they form a BFPM. Since α, x is a BFPM, we only need to check that it has no unnecessarily paced bidders. Suppose some buyer i is spending strictly less than B_i and $\alpha_i < 1$. If i is not tied for any items, then we can increase α_i for some sufficiently small ϵ and retain budget feasibility, contradicting the maximality of α. If i is tied for some item, consider the set $N(i)$ of all bidders tied with i. Now take the transitive closure of this set by repeatedly adding any bidder that is tied with any bidder in $N(i)$. We can now redistribute all the tied items among bidders in $N(i)$ such that no bidder in $N(i)$ is budget constrained (this can be done by slightly increasing i's share of every item they are tied on, then slightly increasing the share of every other buyer in $N(i)$ who is now below budget, and so on). But now there must exist some small enough $\delta > 0$ such that we can increase the pacing multiplier of every bidder in $N(i)$ by δ while retaining budget feasibility and creating no new ties. This contradicts α being maximal. We get that there can be no unnecessarily paced bidders under α.

Finally, to show uniqueness, consider any alternative BFPM α', x'. Consider the set I of buyers such that $\alpha'_i < \alpha_i$; Since $\alpha \geq \alpha'$ and $\alpha \neq \alpha'$ this set must have size at least one. Suppose some buyer in I spends no money under α. Then that buyer is unnecessarily paced, since $\alpha'_i < \alpha_i \leq 1$. Now, suppose the buyers in I all spend money. Then the collective spending of the buyers in I strictly decreases under α'. Since all buyers in I were spending less than their budget under α, and their collective spending strictly decreased, at least one buyer in I must not be spending their whole budget. But $\alpha'_i < \alpha_i \leq 1$ for all $i \in I$, so that buyer must be unnecessarily paced. □

16.3.1 Sensitivity

FPPE enjoys several nice monotonicity and sensitivity properties that SPPE does not. Several of these follow from the maximality property of FPPE: the unique FPPE multipliers α are such that $\alpha \geq \alpha'$ for any other BFPM (α', x').

The following are all guaranteed to weakly increase revenue of the FPPE:

(i) Adding a bidder i: the former FPPE (α, x) is still a BFPM by setting $\alpha_i = 0, x_i = 0$. By α monotonicity prices increase weakly.

(ii) Adding an item: The new FPPE α' satisfies $\alpha' \leq \alpha$ (for contradiction, consider the set of bidders whose multipliers increased, since they win weakly more and prices went up, somebody must break their budget). Now consider the bidders such that $\alpha'_i < \alpha_i$. Those bidders spend their whole budget by the FPPE "no unnecessary pacing" condition. For bidders such that $\alpha'_i = \alpha_i$, they pay the same as before, and win weakly more.

(iii) Increasing a bidder i's budget: the former FPPE (α, x) is still BFPM, so this follows by α maximality.

It is also possible to show that revenue enjoys a Lipschitz property: increasing a single buyer's budget by Δ increases revenue by at most Δ. Similarly, social welfare can be bounded in terms of Δ, though multiplicatively, and it does not satisfy monotonicity.

16.3.2 Convex Program

Next we consider how to compute an FPPE. This turns out to be easier than for SPPE. This is due to a direct relationship between FPPE and market equilibrium: FPPE solutions are exactly the set of solutions to the *quasi-linear* variant of the Eisenberg-Gale convex program for computing a market equilibrium:

$$\begin{aligned} \max_{x\geq 0,\delta\geq 0,u} \quad & \sum_i B_i \log(u_i) - \delta_i \\ u_i \leq & \sum_{j\in[m]} x_{ij}v_{ij} + \delta_i, \forall i \qquad (16.1) \\ & \sum_{i\in[n]} x_{ij} \leq 1, \forall j, \qquad (16.2) \end{aligned} \qquad \begin{aligned} \min_{p\geq 0,\beta\geq 0} \quad & \sum_j p_j - \sum_{i\in[n]} B_i \log(\beta_i) \\ & p_j \geq v_{ij}\beta_i, \forall i \\ & \beta_i \leq 1. \end{aligned} \qquad (16.3)$$

On the left is shown the primal convex program, and on the right is shown the dual convex program. The variables x_{ij} denote the amount of item j that bidder i wins. The leftover budget is denoted by δ_i, it arises from the dual program: it is the primal variable for the dual constraint $\beta_i \leq 1$, which constrains bidder i to paying at most a price-per-utility rate of 1.

The dual variables β_i, p_j correspond to constraints (16.1) and (16.2), respectively. They can be interpreted as follows: β_i is the inverse bang-per-buck: $\min_{j s.t. x_{ij}>0} \frac{p_j}{v_{ij}}$ for buyer i, and p_j is the price of good j.

We can use KKT conditions (see Theorem A.4) to show that FPPE and EG are equivalent. Informally, the correspondence between FPPE and solutions to the convex program follows because β_i specifies a single price-per-utility rate per bidder which exactly yields the pacing multiplier $\alpha_i = \beta_i$. Complementary slackness then guarantees that if $p_j > v_{ij}\beta_i$ then $x_{ij} = 0$, so any item allocated to i has exactly rate β_i. Similarly, complementary slackness on $\beta_i \leq 1$ and the associated primal variable δ_i guarantees that bidder i is only paced if they spend their whole budget.

Theorem 16.7 *An optimal solution to the quasi-linear Eisenberg-Gale convex program corresponds to an FPPE with pacing multiplier $\alpha_i = \beta_i$ and allocation x_{ij}, and vice versa.*

Proof Clearly the quasi-linear Eisenberg-Gale convex program satisfies the Slater constraint qualification: it is satisfies by the proportional allocation where every buyer gets $\frac{1}{n}$ of every item. Thus the optimal solution must satisfy the following KKT conditions:

(i) $\frac{B_i}{u_i} = \beta_i \Leftrightarrow u_i = \frac{B_i}{\beta_i}$,

(ii) $\beta_i \leq 1$,

(iii) $\beta_i \leq \frac{p_j}{v_{ij}}$,

(iv) $x_{ij}, \delta_i, \beta_i, p_j \geq 0$,

(v) $p_j > 0 \Rightarrow \sum_{i\in[n]} x_{ij} = 1$, (vii) $x_{ij} > 0 \Rightarrow \beta_i = \frac{p_j}{v_{ij}}$.
(vi) $\delta_i > 0 \Rightarrow \beta_i = 1$,

It is easy to see that x_{ij} is a valid allocation: the primal program has the exact packing constraints. Budgets are also satisfied (here we may assume $u_i > 0$ since otherwise budgets are satisfied since the bidder wins no items): by KKT condition (i) and KKT condition (vii) we have that for any item j that bidder i is allocated part of:

$$\frac{B_i}{u_i} = \frac{p_j}{v_{ij}} \Rightarrow \frac{B_i v_{ij} x_{ij}}{u_i} = p_j x_{ij}.$$

If $\delta_i = 0$ then summing over all j gives

$$\sum_{j\in[m]} p_j x_{ij} = B_i \frac{\sum_{j\in[m]} v_{ij} x_{ij}}{u_i} = B_i.$$

This part of the budget argument is exactly the same as for the standard Eisenberg-Gale proof in Theorem 11.4. Note that (16.1) always holds exactly since the objective is strictly increasing in u_i. Thus $\delta_i = 0$ denotes full budget expenditure. If $\delta_i > 0$ then (16.1) implies that $u_i > \sum_{j\in[m]} v_{ij} x_{ij}$ which gives:

$$\sum_{j\in[m]} p_j x_{ij} = B_i \frac{\sum_{j\in[m]} v_{ij} x_{ij}}{u_i} < B_i.$$

This shows that $\delta_i > 0$ denotes some leftover budget.

If bidder i is winning some of item j ($x_{ij} > 0$) then KKT condition (vii) implies that the price on item j is $\alpha_i v_{ij}$, so bidder i is paying their bid, as is necessary in a first-price auction. Bidder i is also guaranteed to be among the highest bids for item j: KKT conditions (vii) and (iii) guarantee $\alpha_i v_{ij} = p_j \geq \alpha_{i'} v_{i'j}$ for all i'.

Finally, each bidder either spends their entire budget or is unpaced: KKT condition (vi) says that if $\delta_i > 0$ (that is, some budget is leftover) then $\beta_i = \alpha_i = 1$, so the bidder is unpaced.

Now we show that any FPPE satisfies the KKT conditions for EG. We set $\beta_i = \alpha_i$ and use the allocation x from the FPPE. We set $\delta_i = 0$ if $\alpha < 1$, otherwise we set it to $B_i - \sum_j x_{ij} v_{ij}$. We set u_i equal to the utility of each bidder. KKT condition (i) is satisfied since each bidder either gets a utility rate of 1 if they are unpaced and so $u_i = B_i$, or their utility rate is α_i, so they spend their entire budget for utility B_i/α_i. KKT condition (ii) is satisfies since $\alpha_i \in [0, 1]$. KKT condition (iii) is satisfied since each item bidder i wins has price-per-utility $\alpha_i = \frac{p_j}{v_{ij}} = \beta_i$, and every other item has a higher price-per-utility. KKT conditions ((iv)) and ((v)) are trivially satisfied by the definition of FPPE. KKT condition (vi) is satisfied by our solution construction. KKT condition (vii) is satisfied because a bidder i being allocated any amount of item j means that they have a winning bid, and their bid is equal to $v_{ij}\alpha_i$. □

It follows that an FPPE can be computed in polynomial time (e.g. via the ellipsoid method). Moreover, we can apply various first-order methods to compute large-scale FPPE. For example, the proportional response dynamics can be extended to the FPPE setting.

16.4 In what sense are we in equilibrium?

We introduced the pacing equilibrium using terminology similar to how we previously discussed game-theoretic equilibria such as Nash equilibria. Yet, it is useful to take a moment to consider what the actual equilibrium properties that we are getting under pacing equilibria are. When we defined pacing equilibria, we asked for a certain complementarity condition on the pacing multipliers, the "no unnecessary pacing" condition (Definition 16.5). This condition is not a game-theoretic equilibrium condition, but rather a condition on the budget-management algorithms that the buyers are using. In particular, it is a condition that an online learning algorithm on the Lagrange multiplier of the budget constraint would try to maintain. Now, assuming that you are in a static environment where at each time step, the items from the pacing model show up, then a pacing equilibrium would be stable, in the sense that if everyone bids according to the computed multipliers, and tied goods are split according to the fractional amounts from the equilibrium, then "no unnecessary pacing" is satisfied, so the budget-management algorithms won't change their pacing multipliers. In this sense we are in equilibrium.

However, from the perspective of the buyers, they may or may not be best responding to each other. In the context of *second-price* pacing equilibrium, it is possible to show that a pacing equilibrium is a pure Nash equilibrium of a game where each buyer is choosing their pacing multiplier, and observing their quasi-linear utility (with $-\infty$ utility for breaking the budget). Moreover, in the second-price setting, if we fix the bids of every other buyer, then a pacing multiplier α_i that satisfies no unnecessary pacing is actually a best response over the set of all possible ways to bid in each individual auction. In the case of *first-price* pacing equilibrium, we do not have this property: a buyer might wish to shade their own price in FPPE. In that case, FPPE should be thought of only as a budget-management equilibrium among the algorithmic proxy bidders that control budget expenditure. Secondly, due to this shading, the values v_{ij} that we took as input to the FPPE problem should probably be thought of as the *bids* of the buyers, which would generally be lower than their true values.

Thirdly, we may think of the FPPE concept as a form of market equilibrium, rather than game-theoretic equilibrium. That FPPE corresponds to a market equilibrium under a quasilinear utility model can be shown straightforwardly using the EG result in Theorem 16.7 using largely the same steps as in the proof of Theorem 11.4.

16.5 Conclusion

There are interesting differences in the properties satisfied by SPPE and FPPE. We summarize them quickly here (these are all covered in the literature noted in the Historical Notes):

- FPPE is unique (this can be shown from the convex program, or directly from the monotonicity property of BFPM), SPPE is not.
- FPPE can be computed in polynomial time, computing an SPPE is a PPAD-complete problem.
- FPPE is less sensitive to perturbation (e.g. revenue increases smoothly as budgets are increased).
- SPPE corresponds to a pure-strategy Nash equilibrium, and thus buyers are best responding to each other.
- Both correspond to different market equilibria (but SPPE requires buyer demands to be "supply aware").
- Neither of them are incentive compatible.
- Due to the market equilibrium connection, both can be shown incentive compatible in an appropriate "large market" sense.

FPPE and SPPE have also been studied experimentally, both via random instances, and instances generated from real ad auction data. The most interesting takeaways from those experiments are:

- In practice SPPE multiplicity seems to be very rare
- Manipulation is hard in both SPPE and FPPE if you can only lie about your value per click
- FPPE dominates SPPE on revenue
- Social welfare can be higher in either FPPE or SPPE. Experimentally it seems to largely be a toss-up on which solution concept has higher social welfare.

16.6 Historical Notes

The multiplicative pacing equilibrium results shown in this chapter were developed by Conitzer et al. (2018) for second-price auction markets, and Conitzer et al. (2019) for first-price auction markets. Another strand of literature has studied models where items arrive stochastically and valuations are then drawn independently. Balseiro et al. (2021) show existence of pacing equilibrium for multiplicative pacing as well as several other pacing rules for such a setting; they also give a very interesting comparison of revenue and social welfare properties of the various pacing options in the unique symmetric equilibrium of their setting. Most notably, multiplicative pacing achieves strong social welfare properties, while probabilistic pacing achieves higher revenue properties. Balseiro et al. (2015) show that when bidders get to select their bids individually, multiplicative pacing equilibrium arises naturally via Lagrangian duality on the budget constraint, under a fluid-based mean-field market model. The PPAD-completeness of computing an SPPE was given by Chen et al. (2021a)

The quasi-linear variant of Eisenberg-Gale was given by Chen et al. (2007) and independently by Cole et al. (2017) (an unpublished note from one of the authors in Cole et al. (2017) was in existence around a decade before the publication of Cole et al. (2017)).

The fixed-point theorem that is invoked to guarantee existence of a pure-strategy Nash equilibrium in the smoothed game is by Debreu (1952), Glicksberg (1952), and Fan (1952).

Further reading.

The two papers introducing the SPPE and FPPE models are a good starting point (Conitzer et al., 2022a,b). Balseiro et al. (2021) is a good reference for alternative budget management strategies such as probabilistic throttling. Chen et al. (2021b) is a good reference for probabilistic throttling studied in a setting similar to the SPPE and FPPE models that we studied.

Beyond auctions with budget constraints, there is a broader literature on what is referred to as *autobidding*. Autobidding refers to the broader paradigm of online advertising auctions moving towards a setup where each advertiser offloads their bidding to some form of autobidder (i.e. an algorithm or AI), and the autobidder takes care of optimizing the bids in individual auctions while maintaining constraints expressed by the advertiser, including budget constraints, but also other constraints such as certain return-on-investment constraints. Aggarwal et al. (2024) is a good entry point into the broader autobidding literature. Chapter 17 will discuss some algorithmic details on how an individual autobidder can be implemented via online learning.

17
Pacing Algorithms for Budget Management

In the previous chapter we studied auctions with budgets and repeated auctions. However, we ignored one important aspect: time. In this chapter we consider an auction market setting where a buyer is trying to adaptively pace their bids over time. The goal is to hit the "right" pacing multiplier as before, but each bidder has to learn that multiplier as the market plays out. We will show how to approach this problem using ideas from regret minimization.

17.1 Online Resource Allocation and Second-Price Auctions

Suppose we have a sequence of second-price auctions happening at time steps $t = 1, \dots, T$, and a buyer that is trying to decide how to allocate their budget B across the T time steps. At each time t, the buyer observes their value v_t, and must then submit a bid b_t to the auction. After submitting their bid, they observe whether they won or not, and the price they paid. Let p_t be the highest bid submitted by any other buyer (The notation p_t intentionally suggests a *price*, since we can think of p_t as the price the buyer pays in the event that they win auction t). The algorithm that we will devise does not require the buyer to observe p_t; the algorithm will only require learning the price paid when the buyer wins an auction.

Suppose that the buyer has the benefit of hindsight, and knows the "prices" p_t for each t. Then, if the buyer has a quasilinear utility function, the hindsight optimization problem simply chooses the optimal subset of items to win while satisfying the budget constraint, which is a knapsack problem. Formally, the hindsight optimization problem is the following problem, where $x_t \in \{0, 1\}$

denotes whether to get item t:

$$\max_{x \in \{0,1\}^T} \sum_{t \in [T]} x_t (v_t - p_t) \quad \text{s.t.} \sum_{t \in [T]} x_t p_t \le B. \tag{17.1}$$

17.1.1 Pacing Structure of the Hindsight Optimum

Suppose that we relax Eq. (17.1) to allow for fractional allocation rather than integral allocation. Note that this can only improve the objective value. In that case, we know from strong duality that we can equivalently minimize the Lagrangian

$$\min_{\mu \ge 0} \max_{x \in [0,1]^T} \sum_{t \in [T]} x_t \left(v_t - p_t\right) - \mu \left(\sum_{t \in [T]} x_t p_t - B \right).$$

If we collect all terms that depend on p_t, we get the following:

$$\min_{\mu \ge 0} \max_{x \in [0,1]^T} \sum_{t \in [T]} x_t \left(v_t - (1+\mu) p_t\right) + \mu B.$$

From here, we see that for a fixed μ, x_t should be one exactly when $v_t \ge (1 + \mu)p_t$, and zero otherwise. If we divide through by $1 + \mu$ we get the following condition for when x_t should be one:

$$\frac{v_t}{1+\mu} \ge p_t. \tag{17.2}$$

Notice that this is exactly what pacing achieves: if we set the pacing multiplier $\alpha = 1/(1+\mu)$ and bid according to $b_t = \alpha v_t$ in each auction t then we implement the rule, since we win exactly when Eq. (17.2) is satisfied, without needing to observe the price p_t. Thus, if we know the optimal Lagrange multiplier μ^*, then we can implement the hindsight optimal solution via paced bidding in second-price auctions (modulo tie breaking for auctions t such that $v_t/(1+\mu) = p_t$).

One way to think about why this works is as follows: the second-price auction is truthful, and so we do not need to worry about knowing the competing bids, we only need to bid our truthful valuation. We can think of $\frac{1}{1+\mu} v_t$ as our "budget-adjusted" truthful valuation for each auction t.

Now let us go back to the discrete setting where $x_t \in \{0, 1\}$ for all t, in which case we no longer have strong duality. We will present a more general result that works for *online resource allocation problems*, of which repeated second-price auctions with budgets are a special case. We will see that even with weak duality, we can achieve very nice guarantees, at least under well-behaved inputs (such as stationary stochastic inputs).

17.2 Online Resource Allocation

In the online resource allocation (ORA) problem, we have a set of m resources and a vector $B \in \mathbb{R}^m_{\geq 0}$ specifying our supply of each resource. We receive a sequence of *requests* $\gamma_t = (f_t, b_t, \mathcal{X}_t)$ for each time step t, where $\mathcal{X}_t$ is a decision set, $f_t : \mathcal{X}_t \to [0, \bar{f}]$ is a bounded reward function, and $b_t : \mathcal{X}_t \to [0, \bar{b}]^m$ is a bounded function specifying how much of each resource a given decision consumes. It is assumed that we observe the entire request γ_t before making a decision, and that there is always a *null action* $x_\emptyset$ such that $f_t(x_\emptyset) = b_t(x_\emptyset) = 0$. It will be convenient to denote by $\rho = B/T$ the per-round expenditure target.

The ORA setting is sometimes referred to as a "packing" setting, because the resource consumption constraints always have non-negative consumption, and each constraint is a less-than-or-equals constraint. The case with general constraints will briefly be discussed in Section 17.5.2, where we discuss how to handle constraints such as *return-on-investment* (ROI) constraints.

Example 17.1 (Pacing in Second-Price Auctions) In the case of repeated second-price auctions with budgets, we have that a request corresponds to an auction. The decision set is $\mathcal{X}_t = \{0, 1\}$, the reward is $f_t(x_t) = (v_t - p_t)x_t$, and the resource consumption function is $b_t(x_t) = p_t x_t$. In the second-price setting the buyer does not observe the price p_t ahead of time, and thus they technically do not observe the entire request γ_t. However, we will see that the algorithm only needs to be able to choose the optimal solution $\arg\max_{x_t \in \mathcal{X}_t} x_t\,(v_t - (1-\mu)p_t)$ for an arbitrary $\mu \geq 0$ in order to function, and we saw how to do that via paced bidding in Eq. (17.2).

Now we can generalize the hindsight optimization problem from Eq. (17.1) to the online resource allocation problem as follows:

$$\begin{aligned} \mathrm{OPT}(\vec{\gamma}) := \max_{\{x_t \in \mathcal{X}_t\}_{t=1}^T} & \sum_{t \in [T]} f_t(x_t) \\ \text{s.t.} & \sum_{t \in [T]} b_t(x_t) \leq B. \end{aligned} \tag{17.3}$$

Similarly to the case of Eq. (17.2), we wish to work with the dual of Eq. (17.3). If we Lagrangify the resource constraints then the problem decomposes into

Algorithm 2 Dual Mirror Descent (DMD) for Online Resource Allocation

Input: Total time steps T, total budget B, stepsize $\eta > 0$, Bregman divergence D derived from a 1-strongly convex DGF d, and initial Lagrange multiplier vector λ.

Set $B_1 = B$, $\rho = B/T$.

for $t = 1, \ldots, T$ **do**

See request $\gamma_t = (f_t, b_t, \mathcal{X}_t)$.

Compute decision x_t:

$$\tilde{x}_t = \arg\max_{x \in \mathcal{X}_t} \; f_t(x) - \langle \lambda_t, b_t(x) \rangle,$$

$$x_t = \begin{cases} \tilde{x}_t & \text{if } b_t(\tilde{x}_t) \le B_t \\ x_\emptyset & \text{otherwise} \end{cases}$$

Update remaining budget: $B_{t+1} = B_t - b_t(x_t)$.

Obtain dual subgradient $g_t = \rho - b_t(\tilde{x}_t)$

Update dual variables with OMD:

$$\lambda_{t+1} = \arg\min_{\lambda \in \mathbb{R}^m_{\ge 0}} \eta \langle g_t, \lambda \rangle + D(\lambda \| \lambda_t)$$

---

per-time-period problems, and we get the following problem

$$\min_{\lambda \le 0} \sum_{t \in [T]} \max_{x_t \in \mathcal{X}_t} \left[f_t(x_t) - \langle \lambda, b_t(x_t) \rangle \right] + \langle \lambda, B \rangle.$$

Define the conjugate-like function

$$f_t^*(\lambda) = \max_{x_t \in \mathcal{X}_t} f_t(x_t) - \langle \lambda, b_t(x_t) \rangle.$$

Now use $B = \rho T$ to move $\langle \lambda, B \rangle$ into the parentheses and define the dual function

$$D(\lambda | \vec{\gamma}) = \sum_{t \in [T]} \left[f_t^*(\lambda) + \langle \lambda, \rho \rangle \right],$$

By weak duality we have that the dual function upper bounds Eq. (17.3), i.e. $\mathrm{OPT}(\vec{\gamma}) \le D(\lambda | \vec{\gamma})$.

The *dual mirror descent* (DMD) algorithm for online resource allocation is given in Algorithm 2. At each time step t, DMD computes the optimal decision $\tilde{x}_t$ using the Lagrangified problem, and then uses the decision $\tilde{x}_t$ as long as it is budget feasible (one can slightly improve the algorithm by explicitly taking the budget constraint into account when choosing $\tilde{x}_t$; the performance analysis will be robust to such variations). If it is not feasible, then the null action $x_\emptyset$ is used. Then, the remaining budget is updated, and an online mirror descent steps is taken on the dual multiplier vector λ_t.

The DMD algorithm can be thought of as minimizing the dual $D(\lambda|\vec{\gamma})$. In particular, let $D_t(\lambda) = f_t^*(\lambda) + \langle \lambda, \rho \rangle$ be the t'th dual term. Then we have $D(\lambda|\vec{\gamma}) = \sum_{t\in[T]} D_t(\lambda)$. Now, notice that $\tilde{x}_t$ is the maximizer of $f_t^*(\lambda_t)$, and therefore g_t is a subgradient of $D_t(\lambda_t)$ whenever $\tilde{x}_t = x_t$. It follows that the DMD algorithm is running online mirror descent (OMD) on the subgradients of the dual functions. Let $\|\cdot\|_*$ be the dual norm of the norm $\|\cdot\|$ that d is strongly convex with respect to. Now notice that the gradients g_t satisfy $\|g_t\|_* \le \|\rho + \bar{b}\|_*$. The OMD regret guarantee from Theorem 4.6 then gives that for any $\lambda \in \mathbb{R}^m_{\ge 0}$ and $t \in [T]$:

$$\sum_{s=1}^{t} \langle \lambda_s - \lambda, \rho - b_s(x_s) \rangle \le \frac{D(\lambda \| \lambda_1)}{\eta} + \frac{\eta t \|\rho + \bar{b}\|_*^2}{2} \tag{17.4}$$

We now cover the performance guarantees given by DMD for online resource allocation. We will describe guarantees for the case of stochastic input in detail, and give an overview of results for the case of adversarial inputs.

DMD is called a *best-of-both-worlds* algorithm, because it achieves the best possible guarantee both under stochastic and adversarial inputs. In fact, it is described as *best-of-many-worlds*, because it also has optimal guarantees under a variety of nonstationary input models as well, though we will not cover those here. In the next section we show that under stochastic inputs, DMD achieves an $O(\sqrt{T})$ regret guarantee, and in the following section we show that it achieves a constant factor approximation guarantee under adversarial inputs.

17.3 Stochastic Inputs

In the stochastic case, we assume that each request γ_t is sampled i.i.d. from some underlying probability distribution $\mathcal{P}$ which is unknown to the algorithm. We measure regret as the *expected* difference between the hindsight optimum and the rewards achieved by our algorithm, under the worst-case distribution. Formally, this is defined as

$$R_T = \sup_{\mathcal{P}} \mathbb{E}_{\vec{\gamma}\sim\mathcal{P}^T} \left[\mathrm{OPT}(\vec{\gamma}) - \sum_{t\in[T]} f_t(x_t) \right]. \tag{17.5}$$

For each resource $j \in [m]$, we consider that resource *nearly depleted* at the first time step τ_j such that $\sum_{t=1}^{\tau_j} b_{tj}(x_t) + \bar{b} > B_j$. Notice that τ_j is a random variable. At time τ_j, we are potentially at risk of selecting an action that causes us to exceed the budget for resource j, in which case Algorithm 2 chooses the zero action. If a resource never reaches nearly depleted status then $\tau_j = T$. Now consider $\tau = \min_{j\in[m]} \tau_j$, the first time that any resource is nearly depleted. The analysis of the algorithm will hinge on τ: we will analyze the performance of the algorithm up to time τ, and show that the rewards achieved by time τ already achieve a good regret guarantee.

Theorem 17.2 *Algorithm 2 with stepsize $\eta > 0$, initial dual multiplier $\lambda_1 \in \mathbb{R}^m_{\geq 0}$, and strongly convex DGF with dual norm $\|\cdot\|_*$, satisfies*

$$R_T \leq \frac{\bar{f}\bar{b}}{\min_{j\in[m]} \rho_j} + \frac{\Omega}{\eta} + \frac{\eta T \|\rho + \bar{b}\|_*^2}{2},$$

where $\Omega = \max\left\{D(\lambda\|\lambda_1) | \lambda \in \left\{0, (\bar{f}/\rho_1)e_1, \ldots, (\bar{f}/\rho_m)e_m\right\}\right\}$. If we set $\eta = \sqrt{2\Omega}/(\sqrt{T}\|\rho + \bar{b}\|_*)$ then

$$R_T \leq \frac{\bar{f}\bar{b}}{\min_{j\in[m]} \rho_j} + \sqrt{2\Omega T}\|\rho + \bar{b}\|_*.$$

Proof In order to bound the regret, we will first provide upper and lower bounds on the first and second term in the regret from Eq. (17.5), respectively.

Upper-bounding the hindsight optimum. In order to upper-bound the hindsight optimum, we introduce the *expected dual* $\bar{D}(\lambda|\mathcal{P}) = \mathbb{E}_{(f,b)\sim\mathcal{P}}[f^*(\lambda)] + \langle\lambda, \rho\rangle$. Applying weak duality for an arbitrary $\lambda \in \mathbb{R}^m_{\geq 0}$ and using $\mathrm{OPT}(\vec{\gamma}) \leq T\bar{f}$, we have for *any* $\tau \in [T]$ that

$$\begin{aligned}
\mathbb{E}_{\vec{\gamma}\sim\mathcal{P}^T}[\mathrm{OPT}(\vec{\gamma})] &= \frac{\tau}{T}\mathbb{E}_{\vec{\gamma}\sim\mathcal{P}^T}[\mathrm{OPT}(\vec{\gamma})] + \frac{T-\tau}{T}\mathbb{E}_{\vec{\gamma}\sim\mathcal{P}^T}[\mathrm{OPT}(\vec{\gamma})] \\
&\leq \frac{\tau}{T}\mathbb{E}_{\vec{\gamma}\sim\mathcal{P}^T}[D(\lambda|\vec{\gamma})] + (T-\tau)\bar{f} \\
&= \tau\bar{D}(\lambda|\mathcal{P}) + (T-\tau)\bar{f}. \qquad (17.6)
\end{aligned}$$

Lower-bounding algorithm rewards. For the sum of rewards obtained by Algorithm 2, we first consider the expected reward obtained at a given round $t \in [\tau]$. By how we choose x_t, we have

$$f_t(x_t) = f_t^*(\lambda_t) + \langle\lambda_t, b_t(x_t)\rangle.$$

Now consider the expected value of $f_t(x_t)$ conditional on the information $\xi_{t-1} = \{\gamma_1, \ldots, \gamma_{t-1}\}$; note that λ_t is known given ξ_{t-1}. Then we have that the expected value is

$$\begin{aligned}\mathbb{E}\left[f_t(x_t)|\xi_{t-1}\right] =&\mathbb{E}[f_t^*(\lambda_t)] + \langle\lambda_t, \mathbb{E}[b_t(x_t)|\xi_{t-1}] + \rho - \rho\rangle \\ =&\bar{D}(\lambda_t|\mathcal{P}) - \mathbb{E}[\langle\lambda_t, \rho - b_t(x_t)\rangle|\xi_{t-1}].\end{aligned} \tag{17.7}$$

Next we wish to remove the dependence on the conditioning on ξ_{t-1} in the above expression. To do so, we use a common strategy from the online learning literature based on *martingales* (see Appendix B). We apply the same strategy separately for the expenditure term and the reward term. We do the reward term first. Consider the stochastic process

$$Z_t = \sum_{s=1}^{t} \left[f_s(x_s) - \mathbb{E}[f_s(x_s)|\xi_{s-1}]\right].$$

In words, this process is the sum of deviations around the mean reward for each time step. The process Z_t is a martingale sequence, since the expected value of the unknown term for the current time t is always zero. Next, note that the time τ where some resource becomes nearly depleted is a *stopping time* with respect to the martingale sequence.[1] Since τ is bounded, the *optional stopping theorem* (see Theorem B.2) implies that $\mathbb{E}[Z_\tau] = 0$. Expanding the definition of Z_τ, we have

$$\mathbb{E}\left[\sum_{t=1}^{\tau} \left[f_t(x_t) - \mathbb{E}[f_t(x_t)|\xi_{t-1}]\right]\right] = 0 \tag{17.8}$$

$$\Leftrightarrow \mathbb{E}\left[\sum_{t=1}^{\tau} f_t(x_t)\right] = \mathbb{E}\left[\sum_{t=1}^{\tau} \mathbb{E}[f_t(x_t)|\xi_{t-1}]\right] \tag{17.9}$$

Similarly, for the term $\mathbb{E}[\langle\lambda_t, \rho - b_t(x_t)\rangle|\xi_{t-1}]$ we define the martingale sequence

$$Y_t = \sum_{s=1}^{t} \left[\langle\lambda_t, \rho - b_t(x_t)\rangle - \mathbb{E}[\langle\lambda_t, \rho - b_t(x_t)\rangle|\xi_{t-1}]\right].$$

Again, τ is a stopping time for this sequence, and so the optional stopping theorem implies $\mathbb{E}[Y_\tau] = 0$, and hence

$$\mathbb{E}\left[\sum_{t=1}^{\tau}\langle\lambda_t, \rho - b_t(x_t)\rangle\right] = \mathbb{E}\left[\sum_{t=1}^{\tau}\mathbb{E}[\langle\lambda_t, \rho - b_t(x_t)\rangle|\xi_{t-1}]\right]. \tag{17.10}$$

[1] A stopping time for a finite set of random variables $Z_1, \ldots, Z_T$ is a random variable $\tau \in [T]$ such that knowing $Z_1, \ldots, Z_t$ is enough information to determine whether $\tau = t$.

Now sum Eq. (17.7) over $t \in [\tau]$ and apply Eq. (17.9) and Eq. (17.10) to get

$$\mathbb{E}\left[\sum_{t=1}^{\tau} f_t(x_t)\right] = \mathbb{E}\left[\sum_{t=1}^{\tau} \bar{D}(\lambda_t|\mathcal{P})\right] - \mathbb{E}\left[\sum_{t=1}^{\tau}\langle \lambda_t, \rho - b_t(x_t)\rangle\right]$$
$$\geq \mathbb{E}\left[\tau\bar{D}(\bar{\lambda}_\tau|\mathcal{P})\right] - \mathbb{E}\left[\sum_{t=1}^{\tau}\langle \lambda_t, \rho - b_t(x_t)\rangle\right]. \qquad (17.11)$$

The inequality follows by letting $\bar{\lambda}_\tau = \tau^{-1}\sum_{t=1}^{\tau}\lambda_t$ and noting that $\bar{D}(\cdot|\mathcal{P})$ is a convex function.

Combining upper and lower bounds Now we can bound the expected regret of Algorithm 2 by combining Eq. (17.6) and Eq. (17.11) to get

$$R_T \leq \mathbb{E}\left[\tau\bar{D}(\bar{\lambda}_\tau|\mathcal{P}) + (T-\tau)\bar{f} - \tau\bar{D}(\bar{\lambda}_\tau|\mathcal{P}) + \sum_{t=1}^{\tau}\langle \lambda_t, \rho - b_t(x_t)\rangle\right]$$
$$= \mathbb{E}\left[(T-\tau)\bar{f} + \sum_{t=1}^{\tau}\langle \lambda_t, \rho - b_t(x_t)\rangle\right].$$

Next we show how to use the regret guarantees from OMD to obtain the result. Let $E(t,\lambda) = \frac{\eta t\|\bar{b}+\bar{\rho}\|_*^2}{2} + \frac{D(\lambda,\lambda_1)}{\eta}$ be the OMD bound on the dual regret at time t against an arbitrary dual vector λ. Applying Eq. (17.4) and using $E(\tau,\lambda) \leq E(T,\lambda)$, we have

$$R_T \leq \mathbb{E}\Bigg[\underbrace{(T-\tau)\bar{f} + \sum_{t=1}^{\tau}\langle \lambda, \rho - b_t(x_t)\rangle + E(T,\lambda)}_{\clubsuit}\Bigg]$$

If $\tau = T$ then we choose the comparator $\lambda = 0$ for the dual regret to get that the bracketed term $\clubsuit$ is less than $E(T, 0)$.

If $\tau < T$, then there is some nearly-depleted resource $j \in [m]$, meaning that $\sum_{t=1}^{\tau} b_{tj}(x_t) + \bar{b} > B_j = \rho_j T$. Now we use the comparator $\lambda = (\bar{f}/\rho_j)e_j$ (recall that e_j is the j'th unit vector) for the dual regret to get

$$\clubsuit \leq (T-\tau)\bar{f} + \frac{\bar{f}}{\rho_j}\sum_{t=1}^{\tau}\left(\rho_j - b_{tj}(x_t)\right) + E(T,(\bar{f}/\rho_j)e_j)$$
$$\leq (T-\tau)\bar{f} + \frac{\bar{f}}{\rho_j}\left(\bar{b} - (T-\tau)\rho_j\right) + E(T,(\bar{f}/\rho_j)e_j)$$
$$= \frac{\bar{f}\bar{b}}{\rho_j} + E(T,(\bar{f}/\rho_j)e_j)$$
$$\leq \frac{\bar{f}\bar{b}}{\min_{k\in[m]}\rho_k} + \max_{k\in[m]} E(T,(\bar{f}/\rho_k)e_k)$$

Combining the two bounds on ♣ and taking the maximum over the comparators in the two bounds yields the theorem. □

Theorem 17.2 shows that DMD achieves $O(\sqrt{T})$ regret under stochastic inputs if we choose a stepsize on the order of $1/\sqrt{T}$, and it is known that this is the best possible dependence on T that can be achieved without further restrictions on the input (in fact, the $O(\sqrt{T})$ lower bound holds even when the underlying distribution $\mathcal{P}$ is known ahead of time). As mentioned earlier, DMD also achieves compelling regret guarantees under certain forms of nonstationary input. To give a flavor of such results, suppose that we are in the stochastic setting, but an adversary is given a budget of *corruption*, where they can arbitrarily manipulate up to δ time steps (while respecting the upper bounds on rewards and expenditure). In that case, it is possible to show that the guarantee from Theorem 17.2 extends to this case, with an additional additive term in the regret bound which is linear in δ.

17.4 Adversarial Inputs

Suppose now that the inputs are adversarial, meaning that we face a worst-case sequence of requests $\{\gamma_1, \ldots, \gamma_t\}_{t=1}^T$, rather than having each of them sampled from the same underlying distribution $\mathcal{P}$. In the basic online learning setting of Chapter 4 we were able to give sublinear regret guarantees for adversarial input. However, this turns out to be impossible in the online second-price auctions with budgets setting (and thus it is impossible in the more general ORA setting). We will give an example pair of input sequences that demonstrate this impossibility. First, however, let us introduce the alternative regret notion that one can target with DMD in the adversarial setting.

We will use the idea of asymptotic α-competitiveness instead. This notion is similar to our regret definition in Eq. (17.5), except that we scale the cumulative reward obtained by our algorithms by α before comparing to the hindsight optimum:

$$R_T(\vec{\gamma}) = \mathrm{OPT}(\vec{\gamma}) - \alpha \sum_{t\in[T]} f_t(x_t). \tag{17.12}$$

So, for example, if we can show that the above regret notion is guaranteed to be sublinear for $\alpha = 2$, then we have shown that, asymptotically and when averaged over the number of time steps, our algorithm achieves half of the hindsight optimum.

In the online second-price auctions with budgets setting, achieving $\alpha < \bar{b}/\rho$ is impossible, where $\rho = B/T$ is the per-round budget, and $\bar{b}$ is the highest possible competing price p_t that the buyer may end up paying. Notice that $\bar{b} \leq \bar{v}$ in the case where we have an upper bound on the buyer's valuation, since we never bid more than our value, and thus cannot win any auction prices above our valuation. Based on this bound, if our target expenditure ρ is much smaller than our largest possible per-round expenditure, then we cannot expect to do anywhere near as well as the hindsight-optimal strategy.

The general proof is quite involved, but the high-level idea is not too complicated. Here we show the construction for $\bar{b} = 1, \rho = 1/2$, and thus the claim is that $\alpha < \bar{b}/\rho_i = 2$ is unachievable. The impossibility is via a worst-case instance. In this instance, the buyer always has value 1 for every auction, and the highest other bid p_t comes from one of the two following sequences:

$$\vec{p}^1 = (p_h, \ldots, p_h, 1, \ \ldots\ldots \ , 1)$$
$$\vec{p}^2 = (p_h, \ldots, p_h, p_\ell, \ldots, p_\ell)\,,$$

for $\bar{b} = 1 \geq p_h > p_\ell > 0$. In the top sequence, the buyer sees a sequence of high prices p_h for $T/2$ steps, and then sees a sequence of maximal prices for $T/2$ steps. In the bottom sequence, the buyer sees the same sequence of high prices for $T/2$ steps, but then sees a sequence of low prices p_ℓ for $T/2$ steps. The general idea behind this construction is that in the sequence $\vec{p}^1$, the buyer must buy many of the expensive items priced at p_h in order to maximize their utility, since they receive zero utility for winning items with price 1. However, in the sequence $\vec{p}^2$, the buyer must save money so that they can buy the cheaper items priced at p_ℓ.

For the case we consider here, there are $T/2$ high bids p_h (assume T is even for convenience), followed by the remaining $T/2$ bids. Now, we may set $p_h = 2\rho - \epsilon = 1 - \epsilon$ and $p_\ell = 2\rho - k\epsilon = 1 - k\epsilon$, where ϵ and k are constants that can be tuned. For sufficiently small ϵ, the buyer can only afford to buy $T/2$ items total, no matter the combination of items. Furthermore, buying an item at price p_ℓ yields k times as much utility as buying an item at p_h.

In order to achieve at least half of the optimal utility under $\vec{p}^1$, the buyer must purchase at least $T/4$ of the items priced at p_h. Since they don't know whether they are facing sequence $\vec{p}^1$ or $\vec{p}^2$ until after deciding whether to buy at least $T/4$ of the p_h items, this must also occur under $\vec{p}^2$. But then buyer i can at most afford to buy $T/4$ of the items priced at p_ℓ when they find themselves in the $\vec{p}^2$ case. Now for any $\alpha < 2$, we can pick k and ϵ such that achieving $(1/\alpha)\,\mathrm{OPT}(\vec{\gamma})$ requires buying at least $T/4 + 1$ of the p_ℓ items.

It follows that we cannot hope to design an online algorithm that competes with $(1/\alpha)\,\mathrm{OPT}(\vec{\gamma})$ for $\alpha < \bar{b}/\rho$. It turns out that DMD achieves the optimal asymptotic competitive ratio of $\alpha = \bar{b}/\rho$. For the general setting with m resources, we define $\alpha = \max(\bar{b}/\min_{j\in[m]}\rho_j, 1)$.

Theorem 17.3 *For any sequence of requests $\vec{\gamma}$, the reward obtained by DMD satisfies*

$$\mathrm{OPT}(\vec{\gamma}) - \alpha \sum_{t\in[T]} f_t(x_t) \leq \frac{\bar{f}\bar{b}}{\min_{j\in[m]\rho_j}} + \frac{\alpha\Omega}{\eta} + \frac{\eta T\alpha\|\bar{b}+\rho\|_*^2}{2},$$

where $\Omega = \max\left\{D(\lambda\|\lambda_1)|\lambda \in \{0, (\bar{f}/\alpha\rho_1)e_1, \ldots, (\bar{f}/\alpha\rho_m)e_m\right\}$ *and* $\alpha = \max(\bar{b}/\min_{j\in[m]}\rho_j, 1)$. *If we set* $\eta = \sqrt{2\Omega}/(\sqrt{T}\|\rho + \bar{b}\|_*)$ *then*

$$\mathrm{OPT}(\vec{\gamma}) - \alpha \sum_{t\in[T]} f_t(x_t) \leq \frac{\bar{f}\bar{b}}{\min_{j\in[m]}\rho_j} + \alpha\sqrt{2\Omega T}\|\rho + \bar{b}\|_*.$$

Proof The proof of this result works directly in the primal space, rather than going through the dual as in the stochastic case. We let $\tau \in [T]$ be the stopping time of the algorithm again. Let $x_1^*, \ldots, x_T^*$ be an optimal solution to $\mathrm{OPT}(\vec{\gamma})$. For each $t \in [\tau]$, by our choice of x_t we have $f_t(x_t) \geq f_t(x_t^*) - \langle\lambda_t, b_t(x_t^*) - b_t(x_t)\rangle$ and $0 = f_t(x_\emptyset) \leq f_t(x_t) - \langle\lambda_t, b_t(x_t)\rangle$. It follows that we have

$$\begin{aligned}\alpha f_t(x_t) =& f_t(x_t) + (\alpha - 1)f_t(x_t)\\ \geq& f_t(x_t^*) - \langle\lambda_t, b_t(x_t^*) - b_t(x_t)\rangle + (\alpha-1)\langle\lambda_t, b_t(x_t)\rangle\\ =& f_t(x_t^*) - \langle\lambda_t, b_t(x_t^*) - \alpha(\rho - b_t(x_t)) + \alpha\rho\rangle\\ \geq& f_t(x_t^*) - \langle\lambda_t, \alpha(\rho - b_t(x_t))\rangle,\end{aligned} \tag{17.13}$$

where the last inequality used $\alpha\langle\lambda_t, \rho\rangle \geq \langle\lambda_t, b_t(x_t^*)\rangle$ by the definition of α.

Now, we have that the regret satisfies

$$\begin{aligned}\mathrm{OPT}(\vec{\gamma}) - \alpha \sum_{t\in[T]} f_t(x_t) \leq& \sum_{t\in[T]} f_t(x_t^*) - \alpha \sum_{t\in[\tau]} f_t(x_t)\\ \leq& \sum_{t=\tau+1}^{T} f_t(x_t^*) + \alpha \sum_{t\in[\tau]} \langle\lambda_t, \rho - b_t(x_t)\rangle\\ \leq& (T-\tau)\bar{f} + \alpha \sum_{t\in[\tau]} \langle\lambda, \rho - b_t(x_t)\rangle + \alpha E(T, \lambda),\end{aligned} \tag{17.14}$$

where we used nonnegativity in the first inequality, Eq. (17.13) in the second inequality, and the regret bound for DMD plus the upper bound on rewards in the third inequality.

Now we can use similar logic as in the stochastic case to finish the proof.

First, if $\tau = T$, then we set $\lambda = 0$ to get the desired bound. If $\tau < T$, then we proceed as in the stochastic case by identifying a resource $j \in [m]$ satisfying $\sum_{t\in[\tau]} b_{tj}(x_t) + \bar{b} \geq B_j = T\rho_j$. Compared to the stochastic case, we scale the comparator by α^{-1} to get $\lambda = (\bar{f}/(\alpha\rho_j))e_j$. Then we get:

$$
\begin{aligned}
Eq.\ (17.14) \leq & (T-\tau)\bar{f} + \frac{\bar{f}}{\rho_j}\sum_{t\in[\tau]}(\rho_j - b_{tj}(x_t)) + \alpha E(T, (\bar{f}/(\alpha\rho_j))e_j) \\
\leq & (T-\tau)\bar{f} + \frac{\bar{f}}{\rho_j}(\bar{b} - (T-\tau)\rho_j) + \alpha E(T, (\bar{f}/(\alpha\rho_j))e_j) \\
\leq & \frac{\bar{f}\bar{b}}{\min_{k\in[m]}\rho_k} + \alpha E(T, (\bar{f}/(\alpha\rho_j))e_j)
\end{aligned}
$$

Taking the maximum over the different possible comparators yields the theorem. □

17.5 Extensions

In this section we show a few extensions of the DMD approach to a variety of settings that include additional complications that arise in practice. We omit proofs for these results, but the reader can refer to the historical notes for the papers where these results are first shown and proved.

17.5.1 Spend Plans

While the guarantees for DMD are compelling in that they show robustness against a variety of input models, the adversarial results do leave something to be desired from a practical standpoint. Suppose that we are in an online second-price auctions with budgets setting, and the time steps in the previous ORA model corresponds to minutes of a given day. In that case, a "request" at a given time step corresponds to the set of all auctions that occur in that particular minute of the day. In such a case, we do not expect the input to be adversarial, but we also do not expect it to be stationary (generally we might expect increased relevant traffic during certain times of day, e.g. around 8 am to 10 am if we are advertising something coffee-related). This traffic pattern is fairly predictable (although day-to-day variation may be significant), and we might hope to do better than the adversarial guarantee by *predicting* how much traffic there will be at each time step. In practice, this predictability is addressed through a *spend plan*. A spend plan breaks up the total budget $B \in \mathbb{R}^m_{\geq 0}$ into a set of per-time-step expenditure targets $\rho^1, \ldots, \rho^T$ such that $\sum_{t\in[T]} \rho^t = B$. The goal for our learning algorithm should then be to track this spend plan.

The DMD algorithm in Algorithm 2 attempts to spend the same amount ρ at every time step. A natural way to incorporate the spend plan is to change the step "Obtain dual subgradient" from $g_t = \rho - b_t(\tilde{x}_t)$ to $g_t = \rho^t - b_t(\tilde{x}_t)$, and keep all other parts of the algorithm the same. Intuitively, this change will make it such that for a time step t where the algorithm wants to spend a lot (or a little) due to the structure of f_t and b_t, we do not penalize such overspending (or underspending) relative to the mean spend ρ, because the predicted spend plan already accounts for such increased (or decreased) spending through the variation in ρ^t.

Suppose now that the input requests $\vec{\gamma}$ are sampled from a set of distributions $\gamma_t \sim \mathcal{P}_t$, where each distribution $\mathcal{P}_t$ can change arbitrarily, subject to the upper bounds on rewards and expenditure. This generalizes the adversarial setting, since we can make each distribution put all of its probability mass on a single request, thereby replicating any adversarial input sequence.

Now we require the hindsight optimum to *mostly* agree with the spending plan. In particular, we redefine $\mathrm{OPT}(\vec{\gamma})$ to take as input a set of nonnegative slack variables $\epsilon_1, \ldots, \epsilon_T$, as follows (note that we now consider in-expectation rewards and expenditures, rather than the actual hindsight optimum):

$$\begin{aligned}\mathrm{OPT}(\vec{\gamma}, \vec{\rho}, \epsilon_1, \ldots, \epsilon_T) := \max_{\{x_t \in \mathcal{X}_t\}_{t=1}^T} & \sum_{t\in[T]} \mathbb{E}_{f_t\sim\mathcal{P}_t}[f_t(x_t)] \\ \text{s.t.} & \sum_{t\in[T]} \mathbb{E}_{b_t\sim\mathcal{P}_t}[b_t(x_t)] \le B, \\ & \mathbb{E}_{b_t\sim\mathcal{P}_t}[b_t(x_t)] \le \rho_t + \epsilon_t, \forall t \in [T].\end{aligned} \tag{17.15}$$

Intuitively, if the optimal strategy, when disregarding the spend plan, has expenditures that are captured well by $\vec{\rho}$ then there will exist a set of small $\epsilon_1, \ldots, \epsilon_T \ge 0$ such that Eq. (17.15) captures the unconstrained optimum.

The regret measure is then the difference between this new notion of hindsight optimum minus the cumulative reward achieved by the algorithm:

$$\mathcal{R}_T := \mathrm{OPT}(\vec{\gamma}, \vec{\rho}, \epsilon) - \sum_{t\in[T]} f_t(x_t).$$

The analysis of DMD (with the updated dual subgradients from the expenditure targets ρ^t) can be extended to show the following high-probability regret bound:

Theorem 17.4 *For any given success probability $\delta \in (0,1)$, DMD guarantees*

$$\mathcal{R}_T \leq \frac{\rho_{\min} + 1 + 2R_T^D}{\rho_{\min}} + \left(8 + \frac{8}{\rho_{\min}}\right)\sqrt{2T\ln(T/\delta)} + \frac{1}{\rho_{\min}} \sum_{t\in[T]} \epsilon_t,$$

where R_T^D is the dual regret bound of online mirror descent on the new subgradients g_t, and $\rho_{\min} = \min_{j\in[m],t\in[T]} \rho_j^t$.

17.5.2 Extension to Return on Investment Constraints

In the ad-auction industry, another widespread constraint expressed by advertisers is the *return on spend* (RoS) constraint, sometimes referred to as *return on investment* or *return on ad spend* as well. For this section, let there be a single resource (the advertiser budget in the case of auctions), so that $b_t(x_t) \in [0,1]$. Then, the advertiser has a RoS parameter $\gamma > 0$, and they require that the total value that they receive is greater than γ times the total cost, i.e. $\sum_{t\in[T]} f_t(x_t) \geq \gamma \sum_{t\in[T]} b_t(x_t)$. In that model, the hindsight problem solved by the advertiser is the following:

$$\begin{aligned} \mathrm{OPT}(\vec{\gamma}) := \max_{\{x_t\in\mathcal{X}_t\}_{t=1}^T} & \sum_{t\in[T]} f_t(x_t) \\ \text{s.t.} & \sum_{t\in[T]} b_t(x_t) \leq B, \\ & \gamma \sum_{t\in[T]} b_t(x_t) \leq \sum_{t\in[T]} f_t(x_t). \end{aligned} \tag{17.16}$$

The DMD framework can be extended to this setting. First, if we introduce dual variables $\lambda \geq 0$ and $\mu \geq 0$, then the Lagrangified hindsight problem is as follows,

$$\max_{\{x_t\in\mathcal{X}_t\}_{t=1}^T} \sum_{t\in[T]} f_t(x_t) - \lambda\left(\sum_{t\in[T]} b_t(x_t) - B\right) - \mu\left(\gamma \sum_{t\in[T]} b_t(x_t) - \sum_{t\in[T]} f_t(x_t)\right). \tag{17.17}$$

To get some intuition, and see how this applies in practice, we go back to the second-price auction setting from Section 17.1, but with an additional RoS constraint. Then we can apply the same tricks as when we derived Eq. (17.2). We let $\mathcal{X}_t = [0,1]$, meaning that x_t is a variable denoting whether the advertiser wins the good at time t, and $b_t(x) = p_t x$ is the price of the good at time t. This corresponds to the second-price auction setting, where $f_t(x) = (v_t - p_t)x$ is the quasilinear utility maximization setting under budget and RoS constraints. For a given pair of estimated dual multipliers λ, μ, the advertiser wants to win the auction (i.e. set $x = 1$) exactly when

$$(1 + \mu) f_t(x) \geq (\lambda + \gamma\mu) p_t \tag{17.18}$$

In the quasilinear case, this means that the advertiser should bid $\frac{1+\mu}{1+\lambda+\gamma\mu} v_t$.

In practice, advertisers often wish to maximize *value* subject to the budget and RoS constraint (as opposed to maximizing quasilinear utility). In that case, we have the same setup as in the preceding paragraph, except $f_t(x) = v_t x$. The advertiser should then bid $\frac{1+\mu}{\lambda+\gamma\mu} v_t$.

In order to learn λ and μ in an online fashion, we can apply the DMD framework in Algorithm 2. Algorithm 2 must be changed as follows: When choosing $\tilde{x}_t$, we must follow Eq. (17.17). When obtaining the dual subgradient, we obtain two gradients, $g_t^\lambda = \rho - b_t(\tilde{x}_t)$ and $g_t^\mu = \gamma f_t(\tilde{x}_t) - b_t(\tilde{x}_t)$. Then, when we apply OMD we solve

$$\lambda_{t+1} = \arg\min_{\lambda \in \mathbb{R}_{\geq 0}} \eta g_t^\lambda \lambda + D_\lambda(\lambda \| \lambda_t), \quad \mu_{t+1} = \arg\min_{\mu \in \mathbb{R}_{\geq 0}} \eta g_t^\mu \mu + D_\mu(\mu \| \mu_t).$$

Assuming that we use a separable DGF such as the Euclidean DGF, the entropy DGF, or the logarithm DGF, we can treat these two separate OMD updates as running a single instance of OMD on the two-dimensional decision set $(\lambda, \mu) \in \mathbb{R}^2_{\geq 0}$, with the corresponding Bregman divergence for $\mathbb{R}^2_{\geq 0}$.

What type of guarantees can DMD achieve for this setting? The first thing to note is that we cannot hope to maintain feasibility of the RoS constraint; we may end up violating the RoS constraint on the very first iteration! Instead, what we can hope to show is that the *average* violation over time tends to zero. Secondly, it turns out that the specifics of the DGF choice for OMD become more important. If one uses the Euclidean DGF, then it is possible to show that under stochastic input, DMD achieves $O(\log(T)\sqrt{T})$ regret, while achieving a violation of the RoS constraint also on the order of $O(\log(T)\sqrt{T})$, and thus the average violation across time decreases at a rate of $O(\log T/\sqrt{T})$. See the historical notes for pointers to the literature on these results.

17.6 Historical Notes

The presentation that we give in Section 17.2 to Section 17.3 builds off of Balseiro et al. (2022) who introduced best-of-many-worlds guarantees for DMD, and showed the result for online resource allocation with non-convex decision sets and non-concave reward functions. The proofs of Theorem 17.2 and Theorem 17.3 follow Balseiro et al. (2022), with some reorganization of the proof of Theorem 17.2. The counterexample for the adversarial input setting was given by Balseiro and Gur (2019), who also introduced a DMD algorithm specifically in the context of repeated second-price auctions with a budget constraint, showed no-regret guarantees under stochastic input, and an asymptotically-optimal competitive ratio under adversarial input. Dual-based schemes had been studied extensively prior to that for online linear programming, a special case of online resource allocation, e.g. in Agrawal et al. (2014); Gupta and Molinaro (2016); Devanur et al. (2019). Beyond auction markets, the idea of using paced bids based on the Lagrange multiplier μ has been studied in the revenue management literature, see e.g. Talluri and Van Ryzin (1998), where it is shown that this scheme is asymptotically optimal as T tends to infinity. There is also recent work on the adaptive bidding problem using multi-armed bandits (Flajolet and Jaillet, 2017).

There are a few different papers that deal with the general issue of how to allocate the spending of DMD non-uniformly across the time steps. Jiang et al. (2025) study a nonstationary setting where there is variability in the distributions $\mathcal{P}_t$, but a prior estimate $\tilde{\mathcal{P}}_t$ for each distribution is supplied to the algorithm. This is a stronger form of prior information than the spend plan we describe in Section 17.5.1. Balseiro et al. (2023a) study a model where there is adversarial uncertainty about the total number of time steps that will occur, called *horizon uncertainty*, and develop approaches for constructing optimal spending plans under such horizon uncertainty. Balseiro et al. (2023b) focus on a nonstationary setting like the one in Section 17.5.1, but where we receive a single sample from each $\mathcal{P}_t$, and they show how to construct spend plans using such single-sample information, in order to achieve regret bounds. Stradi et al. (2025) develop the approach presented in Section 17.5.1 where the underlying benchmark is constrained to approximately follow the spend plan, and show no-regret guarantees for a broader framework that generalizes DMD, and also show similar results for the online learning and bandit settings where the request γ_t is not observed before choosing x_t. Note that Jiang et al. (2025) appeared before all the other works in this paragraph, but the publication times are inverted due to differences in how rapidly conferences and journals publish results.

Further reading.

Balseiro et al. (2022) is a good starting point, the paper is very well-written, and the authors provide a nice overview of the proof techniques that are used. For the setting with spend plans, Balseiro et al. (2023b) is a good starting point. For settings where the requests are not observed ahead of time, the reader can refer to Celli et al. (2022) for best-of-many-worlds results similar to those of DMD in the ORA setting. For bidding in first-price auctions with budget constraints, the reader can refer to Wang et al. (2023) or Castiglioni et al. (2024). The latter also handles return-on-spend constraints, a form of non-packing constraint. For bidding in either first-price or second-price auctions with both budget and RoS constraints, in addition to Castiglioni et al. (2024), the reader can also consult Aggarwal et al. (2025).

18
Demographic Fairness

This chapter studies the issue of *demographic fairness*. Demographic fairness is a distinct fairness topic from the earlier fairness topics studied in Chapters 11 and 13 such as envy-freeness and proportionality. Those topics focused on *individual fairness* guarantees. Moreover, in the context of ad auctions, those fairness guarantees are with respect to advertisers, since they are the buyers/agents in the market equilibrium model of the ad auction markets. Demographic fairness, on the other hand, is a fairness notion with respect to the users who are being shown the ads. In the context of the Fisher market models we have studied so far, this means that demographic fairness will be a property measured on the item side, since items correspond to ad impressions for particular users. Secondly, some demographic fairness notions will be with respect to groups of users, rather than individual users. A serious concern with internet advertising auctions and recommender systems is that the increased ability to target users based on features could lead to harmful effects on subsets of the population, such as gender or race-based biases in the types of ads or content being shown. We will start by looking at a few real-world examples where notions of demographic fairness were observed to be violated. We will then describe some potential ideas for implementing fairness in the context of Fisher markets and first-price ad auctions. It is important to emphasize that this is an evolving area, and it is not clear that there is a simple answer to the question of how to guarantee certain types of demographic fairness. Moreover, there are trade-offs between various notions, as well as between fairness and other objectives such as revenue or welfare.

Age Discrimination in Job Ads

ProPublica reported in 2017 that many companies were using age as part of their targeting criteria for job ads they were placing on Facebook (Angwin et al., 2016). This included Amazon, Verizon, UPS and Facebook itself. Quoting from the article:

> Verizon placed an ad on Facebook to recruit applicants for a unit focused on financial planning and analysis. The ad showed a smiling, millennial-aged woman seated at a computer and promised that new hires could look forward to a rewarding career in which they would be "more than just a number."
>
> Some relevant numbers were not immediately evident. The promotion was set to run on the Facebook feeds of users 25 to 36 years old who lived in the nation's capital, or had recently visited there, and had demonstrated an interest in finance.

Whether age-based targeting of job ads is illegal was not yet resolved legally, as of the 2017 article. The federal *Age Discrimination in Employment Act* of 1967 prohibits bias against people aged 40 or older both in hiring and employment. Whether the company placing the ad, as well as Facebook, could be held liable for age discrimination was similarly not clear, since the law was written before the internet age and thus was not formulated with this type of media in mind.

Targeting Housing Ads along Racial Boundaries

ProPublica also reported in 2016 on the fact that advertisers had the ability to run ads that exclude certain "ethnic affinities" such as "Hispanic affinity" or "African-American affinity" on Facebook Angwin and Parris Jr. (2016). Since Facebook does not ask users about race, these affinity categories are stand-in estimates based on user interests and behavior. On the benign side, these features can be used to test for example how an ad in Spanish versus English will perform in a Hispanic population. More generally, it can be used as a tool for advertisers to understand how their products are received by different groups.

However, ProPublica reported that they were able to create a (fake) ad for an event related to first-time home buying, where they could use these categories to exclude various ethnic groups from seeing the ad. When it comes to topics such as housing, the *Fair Housing Act* from 1968 made it illegal

> "to make, print, or publish, or cause to be made, printed, or published any notice, statement, or advertisement, with respect to the sale or rental of a dwelling that indicates any preference, limitation, or discrimination based on race, color, religion, sex, handicap, familial status, or national origin."

In other contexts, such as e.g. traditional newspapers, advertisements are reviewed before being accepted to be shown, in order to ensure that they do not violate these laws. However, in the context of online advertising, the process is much more automated and algorithmic, and the targeting criteria are powerful enough that one has to think carefully about what fairness means and how it can be implemented algorithmically.

For the remainder of the chapter, we will operate under the assumption that we wish to ensure various demographic properties of how ads are shown, for ads that are viewed as "sensitive". Beyond employment and housing, another category of ads that are viewed as sensitive are credit opportunities. Again, existing laws that were created prior to the internet disallow discrimination based on demographic properties in lending.

18.1 Disallowing Targeting

If we wish to prohibit the potential discrimination described above, we could introduce a category of "sensitive ads," where we do not allow age, gender, or racial features to be used as a feature. One might naively think that this would work, but unfortunately there are many ways to perform indirect targeting of these categories. For example, zip code can often be a strong proxy for race, and thus care is needed in order to ensure that we do not allow proxy-based targeting of these sensitive features.

Facebook took such an approach in 2019 (Sandberg, 2019), based on a settlement with various civil rights organizations. In that approach, they disallow targeting on age, gender, zip code, and "cultural affinities" for what they categorize as sensitive ads. That categorization includes housing, employment, and credit opportunities.

While this approach ensures that a certain type of discrimination cannot occur, it does not necessarily rule out other forms of biases in how ads are served. For example, other features can be used as proxies for the above features, or competition from non-sensitive ads may case bias in how sensitive ads are shown. We will show an example of this later case in the next section.

18.2 Demographic Fairness Measures

We will now study some explicit quantitative measures of demographic fairness. These can potentially be used to audit whether a given ad or system contains biases, or as guiding measures for how to adaptively change the allocation system in order to ensure unbiasedness.

To make things concrete, suppose we have m users, and a single sensitive ad i. We will assume that each user j is associated with a feature vector w_j of non-sensitive features. Additionally, each user also belongs to one of two demographic groups, A or B, which is considered a sensitive attribute. Let $g_j \in \{A, B\}$ denote this group. We let G_A and G_B be the set of all indices denoting users in group A or group B, respectively. As usual, we will use $x_{ij} \in [0, 1]$ to denote the probability that the ad i is shown to user j.

Statistical Parity

This notion of demographic fairness asks that ad i is shown at an equal rate across the two groups, in the following sense:

$$\frac{1}{|G_A|} \sum_{j \in G_A} x_{ij} = \frac{1}{|G_B|} \sum_{j \in G_B} x_{ij}.$$

This is an aggregate guarantee; individuals in either group have no guarantee of being treated fairly.

Next, let's see an example of how statistical parity could be broken even though targeting by demographic features is disallowed. Suppose that a sensitive ad (say a job ad) wishes to target users in either demographic, and has a value of \$1 per click, with a click-through rate that depends only on w_j and not g_j. Secondly, there's another ad which is not sensitive, which has a value per click of \$2, and click-through rates of 0.1 and 0.6 for groups A and B respectively. Now, the sensitive ad will never be able to win any slots for group B since even with a CTR of 1, their bid will be lower than $0.6 \cdot 2 = 1.2$. As a result, the sensitive ad will be shown only to group A. A concrete example of how this competition-driven form of bias might occur is when the non-sensitive ad is some form of female-focused product such as clothing or make-up.

A potential criticism of this fairness measure is that it does not require the ad to be shown to equally interested users in both groups. Thus, one could for example worry that the ad might end up buying highly relevant slots among one group, and cheap irrelevant slots in the other group in order to satisfy the constraint.

Similar Treatment

Similar treatment (ST) asks for an individual-level fairness guarantee: if two users j and k have the same non-sensitive feature vector $w_j = w_k$, then they should be treated similarly regardless of the value of g_j and g_k. A simple version of this principle for ad auctions could be that we require $x_{ij} = x_{ik}$ whenever $w_j = w_k$. In practice some features are continuous and thus exact equality between w_j and w_k never occurs. More generally, we may want to insist on some form of similar treatment whenever two users satisfy $w_j \approx w_k$, for some appropriate measure of similarity. Suppose we have a measure $d(w_j, w_k)$ that measures similarity between feature vectors. Then, ST can be defined as

$$|x_{ij} - x_{ik}| \leq d(w_j, w_k).$$

With this definition, we are asking for more than just equality when $w_j = w_k$; instead we also ask that the difference between x_{ij} and x_{ik} should decrease smoothly as the non-sensitive feature vectors get closer to each other, as measured by d.

18.3 Fairness Constraints in FPPE via Taxes and Subsidies

Now we study a potential way that we could implement demographic fairness in the context of Fisher markets and first-price ad auctions. Specifically, we will see that the Eisenberg-Gale convex program lets us derive a tax/subsidy scheme for demographic fairness. The high-level idea is that we can consider a more constrained variant of EG for FPPE, where we insist that the computed allocation satisfies our fairness constraints, and then we can use KKT conditions to derive appropriate taxes and subsidies from the resulting Lagrange multipliers on the fairness constraints. To be concrete, suppose that for a group of buyers $I \subset [n]$, perhaps representing a particular group of sensitive ads such as job ads, we wish to enforce statistical parity across this group in an FPPE setting. Then, we can consider the following constrained version of the EG program:

$$\max_{x\geq 0, \delta \geq 0, u} \quad \sum_i B_i \log(u_i) - \delta_i$$

$$u_i \leq \sum_{j\in[m]} x_{ij} v_{ij} + \delta_i, \forall i \tag{18.1}$$

$$\sum_{i\in[n]} x_{ij} \leq 1, \forall j, \tag{18.2}$$

$$\sum_{i\in I} \sum_{j\in G_A} x_{ij} = \sum_{i\in I} \sum_{j\in G_B} x_{ij}. \tag{18.3}$$

Now, our EG program maximizes the quasilinear EG objective, but over a smaller set of feasible allocations: those that satisfy the statistical parity constraint across buyers in I.

The key to analyzing this new quasilinear EG variant is to use the Lagrange multipliers on Eq. (18.3). Let (x, p) be the optimal allocation, and let p be the prices derived from the Lagrange multipliers on the supply constraints Eq. (18.2). Let λ be the Lagrange multiplier on Eq. (18.3). We will show that (x, p, λ) is a form of market equilibrium, where we charge each buyer $i \in I$ a price of $p_j + \lambda$ for $j \in A$ and a price of $p_j - \lambda$ for $j \in B$, where λ is the Lagrange multiplier on Eq. (18.3). Buyers $i \notin I$ are simply charged the price vector p. Clearly, this is not our usual notion of market equilibrium: we are charging two different sets of prices: demographically-adjusted prices for buyers in I and regular prices for buyers not in I.

First, consider some non-sensitive buyer $i \notin I$. For such a buyer, we can show that $x_i \in D_i(p)$ using the exact same argument as in the case of the standard quasilinear EG program in Theorem 16.7. Similarly, we can show that each item is fully allocated if $p_j > 0$ using the same arguments as before. It is also direct from feasibility that the statistical parity constraint is satisfied.

Given the above, we only need to see what happens for buyers $i \in I$. Ignoring feasibility conditions which are straightforward, the KKT conditions pertaining to buyer i are as follows:

(i) $\frac{B_i}{u_i} = \beta_i \Leftrightarrow u_i = \frac{B_i}{\beta_i}$,
(ii) $\beta_i \leq 1$,
(iii) $\beta_i \leq \frac{p_j \pm \lambda}{v_{ij}}$,
(iv) $\delta_i > 0 \Rightarrow \beta_i = 1$,
(v) $x_{ij} > 0 \Rightarrow \beta_i = \frac{p_j \pm \lambda}{v_{ij}}$.

Here, the $\pm$ should be interpreted as '+' for $j \in A$ and '−' for $j \in B$. Now it is straightforward from KKT conditions (iii) and (v) that buyer i buys only items with optimal price-per-utility under the prices $p_j \pm \lambda$. From here, the same argument as in Theorem 16.7 can be performed in order to show that buyer i spends their whole budget, which shows that they received a bundle $x_i \in D_i(p \pm \lambda)$.

It follows from the above that (x, p, λ) is a market equilibrium (with different prices for I and $[n] \setminus I$), and thus we can use the Lagrange multiplier λ as a tax/subsidy scheme in order to enforce statistical parity.

18.4 Historical Notes

The field of "algorithmic fairness" pioneered a lot of the fairness considerations that we considered in this chapter, in the context of machine learning. Dwork et al. (2012) introduced similar treatment in the context of machine learning classification, and the notion that we use here for ad auction allocation is an adaptation of their definitions. They also study statistical parity in the classification context. A book-level treatment of fairness in machine learning is given by Barocas et al. (2019). Many of these fairness notions were also previously known in the education testing and psychometrics literature. See the biographical notes in Barocas et al. (2019) for an overview of these older works. The quasilinear Fisher market model with statistical parity constraints via taxes and subsidies was studied in Peysakhovich et al. (2023), which also studies several other fairness questions in the context of Fisher markets. A related work is

Jalota et al. (2023). This work does not study fairness directly, but shows how per-buyer linear constraints can be implemented similarly to what we describe in Section 18.3.

Further reading.

The book by Barocas et al. (2019) is a good introduction to the field of algorithmic fairness, and contains many references to the literature on fairness in machine learning and recommender systems.

Appendix A
Optimization Background

A.1 Basic Definitions

Definition A.1 A function f is *quasi-concave* if for all x, y and $\lambda \in [0, 1]$ it holds that $f(\lambda x + (1 - \lambda)y) \geq \min(f(x), f(y))$.

Definition A.2 For $\mu \geq 0$, a function f is μ-strongly convex (also stated as "strongly convex with modulus μ") relative to the norm $\|\cdot\|$ if

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2,$$

where $\nabla f(x)$ is an arbitrary subgradient of f at x.

Instead of saying that f is "strongly convex with modulus μ," we often use the shorthand "f is μ-strongly convex."

For a twice differentiable function $f : \mathbb{R}^m \to \mathbb{R}$ with Hessian $\nabla^2 f$, a sufficient, but not necessary, condition for strong convexity of f on a set X relative to a norm $\|\cdot\|$ is the following:

$$h^\top \nabla^2 f(x) h \geq \mu \|h\|^2, \ \forall x \in X, h \in \mathbb{R}^m.$$

The condition is necessary when X is full-dimensional in its ambient space.

A.2 Basic Inequalities

In the following I list a number of inequalities that are used in the book. All but the last inequality are "standard" inequalities and stated without proof. Many have elegant proofs, see e.g. Steele (2004) for a beautiful book on inequalities and their proofs.

The Cauchy-Schwarz inequality states that

$$\langle a, b\rangle \le \|a\|_2\|b\|_2. \tag{A.1}$$

The AM-GM inequality (AM-GM stands for arithmetic-mean geometric-mean) states that

$$\sqrt[n]{x_1 \cdot x_2 \cdots x_n} \le \frac{1}{n}\sum_{i\in[n]} x_i. \tag{A.2}$$

The "Young's inequality with ϵ" or "Peter-Paul inequality" states that for all $\epsilon > 0$,

$$ab \le \frac{a^2}{2\epsilon} + \frac{\epsilon b^2}{2}. \tag{A.3}$$

Euler's formula for complex numbers states

$$e^{ix} = \cos(x) + i\sin(x). \tag{A.4}$$

Suppose we have a set of numbers $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$ such that they are all nonnegative, and $\sum_{i\in[n]} b_i > 0$. Then we have

$$\frac{\sum_{i\in[n]} a_i}{\sum_{i\in[n]} b_i} \ge \min_{i\in[n]} \frac{a_i}{b_i}. \tag{A.5}$$

To see this, let $\rho = \min_{i\in[n]} a_i/b_i$. Then we have $a_i \ge \rho b_i$ for all $i \in [n]$. Summing this inequality gives

$$\sum_{i\in[n]} a_i \ge \rho \sum_{i\in[n]} b_i.$$

Dividing through by $\sum_{i\in[n]} b_i$ yields the inequality.

A.3 Karush-Kuhn-Tucker Conditions

Consider a convex program and its dual

$$\begin{aligned} \min_x \quad & f(x) \\ g_j(x) \le 0,\ & \forall j \in [m] \\ x \ge 0, & \end{aligned} \tag{A.6}$$

$$\begin{aligned} &\max_{\lambda\ge 0} \quad q(\lambda) \\ &q(\lambda) := \min_{x\ge 0} L(x,\lambda) \\ &L(x,\lambda) := f(x) + \sum_{j\in[m]} \lambda_i g_m(x), \end{aligned} \tag{A.7}$$

with Lagrange multipliers λ_i for each constraint i.

To guarantee KKT conditions and Strong duality, we need *Slater's condition*:

there exists some $x \geq 0$ such that $g_j(x) < 0$ for all j, and x is in the domain of f.

First we state a result purely related to the existence of Lagrange multipliers supporting an optimal solution, without reference to the dual problem in Eq. (A.7).

Theorem A.3 *Suppose Slater's condition is satisfied, and let x^* be an optimal solution to Eq.* (A.6). *Then there exists Lagrange multiplier vectors $\lambda \in \mathbb{R}^m_{\geq 0}$ and $\mu \in Rplus^n$ such that:*

(i) *Stationarity:* $0 \in \partial f(x^*) + \sum_{j \in [m]} \lambda_j \partial g_j(x^*) + \mu$.
(ii) *Complementary slackness:* $\lambda_j g_j(x^*) = 0, \forall j \in [m]$ *and* $x^*_i \mu_i = 0, \forall i \in [n]$.

Similarly, under Slater's condition we have strong duality, in which case we get the following primal and dual KKT conditions:

Theorem A.4 *Assume that Slater's condition is satisfied. If* (A.6) *has a finite optimal value f^* then* (A.7) *has a finite optimal value q^* and $f^* = q^*$. Furthermore, a solution pair x^*, λ^* is optimal if and only if the following Karush-Kuhn-Tucker (KKT) conditions hold:*

- (primal feasibility) x^* *is a feasible solution of* (A.6).
- (dual feasibility) $\lambda^* \geq 0$.
- (complementary slackness) $\lambda^*_i g_i(x^*) = 0$ *for all i.*
- (stationarity) $x^* \in \arg\min_{x \geq 0} L(x, \lambda^*)$.

Theorem A.4 is a specialization to settings covered in this book. In reality much stronger statements can be made: For a more general statement of the strong duality theorem and KKT conditions used here, see Bertsekas et al. (2003) Proposition 6.4.4. The KKT conditions can be significantly generalized beyond convex programming. Bauschke and Combettes (2017, Proposition 27.21) also gives a very general KKT theorem.

A.4 Bregman Divergences and Proximal Mappings

Consider the problem

$$\text{prox}(g) = \arg\min_{x \in \mathcal{X}} \langle g, x \rangle + d(x),$$

where $d : \mathcal{X} \to \mathbb{R}$ is a strongly convex function with modulus $\mu > 0$ with respect to a norm $\|\cdot\|$ (see Definition A.1). Let $\|\cdot\|_*$ denote the dual norm. The function $\text{prox}(g)$ has a useful interpretation in terms of the convex conjugate $d^*(g) =$

$\max_x \langle g, x\rangle - d(x)$. Notice that if we change the maximum to a minimum by dividing through by -1, then we get $d^*(g) = -(\min_x \langle -g, x\rangle + d(x))$. We have that $\text{prox}(g)$ is equal to the argument of this minimization problem. Next we note that the gradient $\nabla d^*(g)$ is exactly equal to this argument by Danskin's theorem, due to the strong convexity of d which ensures that there is a unique optimal solution to the minimization problem. It follows that $\text{prox}(g) = \nabla d^*(-g)$.

A basic fact from convex analysis says that if a function d is strongly convex with modulus μ, then the gradient of its convex conjugate is $1/\mu$-Lipschitz. This is formalized below, where we also provide a direct proof.

Lemma A.5 *The prox function satisfies*

$$\|\text{prox}(g) - \text{prox}(\hat{g})\| \leq \frac{1}{\mu}\|g - \hat{g}\|_*.$$

Proof Let $x^* = \text{prox}(g)$ and $\hat{x}^* = \text{prox}(\hat{g})$. Let $f(x) = \langle g, x\rangle + d(x)$ and $\hat{f}(x) = \langle \hat{g}, x\rangle + d(x)$. Since the sum of a linear and strongly convex function is strongly convex with the same modulus, we have that f is strongly convex with modulus μ. Combining strong convexity and optimality of x^* and $\hat{x}^*$, we have that

$$\frac{\mu}{2}\|x^* - \hat{x}^*\|^2 \leq f(\hat{x}^*) - f(x^*) = \langle g, \hat{x}^* - x^*\rangle + d(\hat{x}^*) - d(x^*),$$
$$\frac{\mu}{2}\|x^* - \hat{x}^*\|^2 \leq f(x^*) - f(\hat{x}^*) = \langle \hat{g}, x^* - \hat{x}^*\rangle + d(x^*) - d(\hat{x}^*).$$

Summing the inequalities and applying Hölder's inequality yields

$$\mu\|x^* - \hat{x}^*\|^2 \leq \langle g - \hat{g}, \hat{x}^* - x^*\rangle \leq \|g - \hat{g}\|_*\|x^* - \hat{x}^*\|.$$

□

The Bregman divergence $D(x'\|x) = d(x') - d(x) = \langle \nabla d(x), x' - x\rangle$ (introduced in Chapter 4) is strongly convex as long as the distance-generating function d is strongly convex, with the same modulus. Since d is strongly convex and $D(x'\|x)$ measures the difference between $d(x')$ and the first-order approximation at $d(x)$, we have the inequality

$$D(x'\|x) \geq \|x' - x\|^2, \tag{A.8}$$

where $\|\cdot\|$ is the norm that d is strongly convex with respect to.

A.5 Berge's Maximum Theorem

Berge's maximum theorem is a useful tool from optimization theory which gives conditions under which the solution to a maximization problem is continuous in the parameters of the problem. The theorem is stated below. It is used widely in economics, since the decision problem of an agent in a game or a buyer in a competitive market may face a parameterized maximization problem, e.g. with prices as the parameters in the case of a buyer in a competitive market. Proofs of the results stated here can be found in Sundaram (1996).

Let $\theta \in \Theta$ be the set of parameters that we vary (e.g. the prices that are input to a demand function), and let the optimization variables $x \in \mathcal{X}$. The theorem is concerned with optimization problems of the form

$$\begin{aligned} &\max_x f(x, \theta) \\ &\text{s.t. } x \in X(\theta). \end{aligned} \tag{A.9}$$

Let $f^*(\theta) \in \mathbb{R}$ be the optimal value of the problem for the parameter choice θ, and let $x^*(\theta) \subset \mathcal{X}$ be the set of optimal solutions to the problem. Berge's maximum theorem gives conditions under which these functions are continuous in θ. In order to work with Berge's maximum theorem, we will need the notion of upper hemicontinuity.

Definition A.6 A set-valued mapping $\phi : X \to \mathcal{P}(X)$ is upper hemicontinuous at a point $x \in X$ if for every open set $U \subset X$ containing $\phi(x)$, there exists an open set $V \subset X$ containing x such that for all $y \in V$, $\phi(y) \subseteq U$. A set-valued mapping ϕ is upper hemicontinuous on X if it is upper hemicontinuous at every point in X.

If the correspondence ϕ is compact-valued, then upper hemicontinuity is equivalent to having a *closed graph* (see Section 10.2).

The standard version of the theorem guarantees continuity of the optimal value function $f^*(\theta)$ and upper hemicontinuity of the optimal solution set $x^*(\theta)$ under mild continuity conditions:

Theorem A.7 *Let $f : \mathcal{X} \times \Theta \to \mathbb{R}$ be a continuous function, and let $X(\theta)$ be a nonempty compact set for all $\theta \in \Theta$. If $X(\theta)$ is continuous (i.e. both upper and lower hemicontinuous) in θ, then the optimal value function $f^*(\theta)$ is continuous in θ, and the optimal solution set $x^*(\theta)$ is upper hemicontinuous in θ.*

There is also a stronger version of the theorem which gives additional properties of the optimal solution set $x^*(\theta)$ when the optimization problem is a convex program. We use the same setup as in Theorem A.7, but we assume that f is concave in x for all $\theta \in \Theta$ and the decision set $X(\theta)$ is convex for any θ.

Theorem A.8 *Let $f : \mathcal{X} \times \Theta \to \mathbb{R}$ be a concave continuous function, and let $X(\theta)$ be a nonempty compact convex set for all $\theta \in \Theta$.*

(i) *If f is concave in x for all $\theta \in \Theta$, then $x^*(\theta)$ is a convex-valued correspondence.*
(ii) *If f is strictly concave in x for all $\theta \in \Theta$, then the optimal solution set $x^*(\theta)$ is single-valued and continuous in θ.*

Appendix B
Probability Background

A discrete-time martingale is a stochastic process consisting of a sequence of random variables $Z_1, Z_2, \ldots$ such that for any time t, we have

$$\mathbb{E}[Z_t] < \infty,$$
$$\mathbb{E}[Z_t | Z_1, \ldots, Z_{t-1}] = Z_{t-1}.$$

The first condition is a regularity condition ensuring finiteness. In applications of martingales in online learning this is often satisfied trivially because rewards are usually assumed to be bounded, such as in Section 17.2. The second condition, in words, requires that the expected value of the process at a given time step t (conditional on all past realizations of random variables) is exactly the realized value at time step $t-1$. In applications in online learning, a martingale is often constructed by defining a random variable equal to the sum of deviations around the expected value of some sequence of random variables, e.g. rewards or resource expenditures.

Martingales are often exemplified via gambling: suppose that the stochastic process describes a gambler's wealth over a discrete set of time steps. Then the martingale property ensures that the gambling process is *fair* in the sense that the expected losses and gains cancel out.

A classical result that is used extensively in online learning is the *Azuma-Hoeffding inequality*, which allows one to bound how much a martingale process varies around its initial value. This is frequently used in online learning to bound a *martingale difference sequence* such as the sum of deviations around the mean loss at each iteration t when an online learning algorithm observes unbiased estimates of a loss vector.

Theorem B.1 (Azuma-Hoeffding inequality) *Suppose $Z_0, Z_1, \ldots$ is a martingale such that $|Z_t - Z_{t-1}| \le c_t$ almost surely for all $t \in [T]$. Let $d_t = \sum_{\tau \in [t]} c_\tau^2$.*

Then we have that

$$\mathbb{P}[|X_t - X_0| \geq \epsilon] \leq 2\exp\left(\frac{-\epsilon^2}{2d_t}\right).$$

A random variable $\tau \in \mathbb{Z}_{>0}$ is a *stopping time* with respect to a martingale sequence $Z_1, Z_2, \ldots$ if the question of whether $\tau = t$ for any $t > 0$ is determined purely by the variables up to time t, i.e. the variables $Z_1, \ldots, Z_t$.

We define the *stopped process* $\bar{Z}_t$ as

$$\bar{Z}_t = \begin{cases} Z_t & \text{if } t \leq \tau \\ Z_\tau & \text{if } t > \tau \end{cases}$$

Suppose Z_t is the wealth of a gambler for each round of gambling they participate in. Now suppose that τ is a random variable deciding when the gambler quits the gambling process. In that case, the stopped process $\bar{Z}_t$ tells us the wealth of the gambler when they stop. The optional stopping theorem, shown below, tells us that when Z_t is a martingale, i.e. a fair gambling process, then the expected initial wealth of the gambler equals their expected wealth at the stopped time. Thus, there is no way for the gambler to devise a stopping process such that their expected wealth when they quit is higher (or lower) than their initial wealth.

Theorem B.2 (Optional Stopping Theorem) *Let $Z_1, \ldots$ be a martingale sequence, and let τ be a stopping time. If any of the following conditions hold:*

(i) *The random variables $\bar{Z}_1, \bar{Z}_2, \ldots$ are uniformly bounded,*
(ii) *τ is bounded,*
(iii) *$\mathbb{E}[\tau] < \infty$, and there exists $M < \infty$ such that $\mathbb{E}[|Z_{t+1} - Z_t| Z_1, \ldots, Z_t] < M$,*

then $\mathbb{E}[Z_\tau] = \mathbb{E}[Z_1]$.

Further reading.

The reader is referred to Ross (1995) for an introduction to the tools from stochastic processes used here. For a measure-theoretic introduction, the reader is referred to Schilling (2017).

Bibliography

Abernethy, Jacob, Bartlett, Peter L, and Hazan, Elad. 2011. Blackwell approachability and no-regret learning are equivalent. Pages 27–46 of: *Proceedings of the 24th Annual Conference on Learning Theory*. JMLR Workshop and Conference Proceedings.

Aggarwal, Gagan, Badanidiyuru, Ashwinkumar, Balseiro, Santiago R, Bhawalkar, Kshipra, Deng, Yuan, Feng, Zhe, Goel, Gagan, Liaw, Christopher, Lu, Haihao, Mahdian, Mohammad, et al. 2024. Auto-bidding and auctions in online advertising: A survey. *ACM SIGecom Exchanges*, **22**(1), 159–183.

Aggarwal, Gagan, Fikioris, Giannis, and Zhao, Mingfei. 2025. No-regret algorithms in non-truthful auctions with budget and roi constraints. Pages 1398–1415 of: *Proceedings of the ACM on Web Conference 2025*.

Agrawal, Shipra, Wang, Zizhuo, and Ye, Yinyu. 2014. A dynamic near-optimal algorithm for online linear programming. *Operations Research*, **62**(4), 876–890.

Aliprantis, Charalambos D, and Border, Kim C. 2006. *Infinite dimensional analysis: a hitchhiker's guide*. Springer Science & Business Media.

Angwin, Julia, and Parris Jr., Terry. 2016. Facebook Lets Advertisers Exclude Users by Race. *ProPublica*.

Angwin, Julia, Scheiber, Noam, and Tobin, Ariana. 2016. Dozens of Companies Are Using Facebook to Exclude Older Workers From Job Ads. *ProPublica*.

Arrow, Kenneth J, and Debreu, Gerard. 1954. Existence of an equilibrium for a competitive economy. *Econometrica: Journal of the Econometric Society*, 265–290.

Balseiro, Santiago, Kim, Anthony, Mahdian, Mohammad, and Mirrokni, Vahab. 2021. Budget-Management Strategies in Repeated Auctions. *Operations research*, 859–876.

Balseiro, Santiago, Kroer, Christian, and Kumar, Rachitesh. 2023a. Online resource allocation under horizon uncertainty. Pages 63–64 of: *Abstract Proceedings of the 2023 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*.

Balseiro, Santiago R, and Gur, Yonatan. 2019. Learning in repeated auctions with budgets: Regret minimization and equilibrium. *Management Science*, **65**(9), 3952–3968.

Balseiro, Santiago R, Besbes, Omar, and Weintraub, Gabriel Y. 2015. Repeated auctions with budgets in ad exchanges: Approximations and design. *Management Science*, **61**(4), 864–884.

Balseiro, Santiago R, Lu, Haihao, and Mirrokni, Vahab. 2022. The best of many worlds: Dual mirror descent for online allocation problems. *Operations Research*.

Balseiro, Santiago R, Kumar, Rachitesh, Mirrokni, Vahab, Sivan, Balasubramanian, and Wang, Di. 2023b. Robust budget pacing with a single sample. Pages 1636–1659 of: *International Conference on Machine Learning*. PMLR.

Barman, Siddharth, Krishnamurthy, Sanath Kumar, and Vaish, Rohit. 2018. Finding fair and efficient allocations. Pages 557–574 of: *Proceedings of the 2018 ACM Conference on Economics and Computation*.

Barocas, Solon, Hardt, Moritz, and Narayanan, Arvind. 2019. *Fairness and Machine Learning*. fairmlbook.org. `http://www.fairmlbook.org`.

Bauschke, Heinz H, and Combettes, Patrick L. 2017. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer.

Beck, Amir. 2017. *First-order methods in optimization*. Vol. 25. SIAM.

Beck, Amir, and Teboulle, Marc. 2003. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, **31**(3), 167–175.

Begle, Edward G. 1950. A fixed point theorem. *Annals of Mathematics*, **51**(3), 544–550.

Bei, Xiaohui, Garg, Jugal, and Hoefer, Martin. 2019. Ascending-price algorithms for unknown markets. *ACM Transactions on Algorithms (TALG)*, **15**(3), 1–33.

Bertsekas, Dimitri P, Nedic, A, and Ozdaglar, A. 2003. Convex analysis and optimization. 2003. *Athena Scientific*.

Bertsimas, Dimitris, and Tsitsiklis, John N. 1997. *Introduction to linear optimization*. Vol. 6. Athena scientific Belmont, MA.

Birnbaum, Benjamin, Devanur, Nikhil R, and Xiao, Lin. 2011. Distributed algorithms via gradient descent for fisher markets. Pages 127–136 of: *Proceedings of the 12th ACM conference on Electronic commerce*. ACM.

Blackwell, David. 1956. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, **6**(1), 1–8.

Börgers, Tilman. 2015. *An introduction to the theory of mechanism design*. Oxford university press.

Borgs, Christian, Chayes, Jennifer, Immorlica, Nicole, Jain, Kamal, Etesami, Omid, and Mahdian, Mohammad. 2007. Dynamics of bid optimization in online advertisement auctions. Pages 531–540 of: *Proceedings of the 16th international conference on World Wide Web*.

Bowling, Michael, Burch, Neil, Johanson, Michael, and Tammelin, Oskari. 2015. Heads-up limit hold'em poker is solved. *Science*, **347**(6218), 145–149.

Boyd, Stephen P, and Vandenberghe, Lieven. 2004. *Convex optimization*. Cambridge university press.

Brams, Steven J, and Taylor, Alan D. 1996. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press.

Brown, Matthew, Sinha, Arunesh, Schlenker, Aaron, and Tambe, Milind. 2016. One size does not fit all: A game-theoretic approach for dynamically and effectively screening for threats. In: *Thirtieth AAAI Conference on Artificial Intelligence*.

Brown, Noam, and Sandholm, Tuomas. 2017. Safe and nested subgame solving for imperfect-information games. Pages 689–699 of: *Advances in neural information processing systems*.

Brown, Noam, and Sandholm, Tuomas. 2018. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, **359**(6374), 418–424.

Brown, Noam, and Sandholm, Tuomas. 2019a. Solving imperfect-information games via discounted regret minimization. Pages 1829–1836 of: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33.

Brown, Noam, and Sandholm, Tuomas. 2019b. Superhuman AI for multiplayer poker. *Science*, **365**(6456), 885–890.

Brown, Noam, Lerer, Adam, Gross, Sam, and Sandholm, Tuomas. 2018a. Deep Counterfactual Regret Minimization. *arXiv preprint arXiv:1811.00164*.

Brown, Noam, Sandholm, Tuomas, and Amos, Brandon. 2018b. Depth-limited solving for imperfect-information games. Pages 7663–7674 of: *Advances in Neural Information Processing Systems*.

Bubeck, Sébastien, et al. 2015. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, **8**(3-4), 231–357.

Budish, Eric. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, **119**(6), 1061–1103.

Budish, Eric, Cachon, Gérard P, Kessler, Judd B, and Othman, Abraham. 2016. Course match: A large-scale implementation of approximate competitive equilibrium from equal incomes for combinatorial allocation. *Operations Research*, **65**(2), 314–336.

Burch, Neil. 2018. *Time and space: Why imperfect information games are hard*. Ph.D. thesis, University of Alberta.

Burch, Neil, Johanson, Michael, and Bowling, Michael. 2014. Solving imperfect information games using decomposition. In: *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Burch, Neil, Moravcik, Matej, and Schmid, Martin. 2019. Revisiting cfr+ and alternating updates. *Journal of Artificial Intelligence Research*, **64**, 429–443.

Caragiannis, Ioannis, Kurokawa, David, Moulin, Hervé, Procaccia, Ariel D, Shah, Nisarg, and Wang, Junxing. 2016. The unreasonable fairness of maximum Nash welfare. Pages 305–322 of: *Proceedings of the 2016 ACM Conference on Economics and Computation*. ACM.

Caragiannis, Ioannis, Kurokawa, David, Moulin, Hervé, Procaccia, Ariel D, Shah, Nisarg, and Wang, Junxing. 2019. The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, **7**(3), 1–32.

Castiglioni, Matteo, Celli, Andrea, and Kroer, Christian. 2024. Online Learning under Budget and ROI Constraints via Weak Adaptivity. Pages 5792–5816 of: *International Conference on Machine Learning*. PMLR.

Celli, Andrea, Castiglioni, Matteo, and Kroer, Christian. 2022. Best of many worlds guarantees for online learning with knapsacks. *arXiv preprint arXiv:2202.13710*.

Cesa-Bianchi, Nicolo, and Lugosi, Gábor. 2006. *Prediction, learning, and games*. Cambridge university press.

Chambolle, Antonin, and Pock, Thomas. 2016. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*, **159**(1-2), 253–287.

Chen, Lihua, Ye, Yinyu, and Zhang, Jiawei. 2007. A note on equilibrium pricing as convex optimization. Pages 7–16 of: *International Workshop on Web and Internet Economics*. Springer.

Chen, Xi, and Teng, Shang-Hua. 2009. Spending is not easier than trading: on the computational equivalence of Fisher and Arrow-Debreu equilibria. Pages 647–656 of: *International Symposium on Algorithms and Computation*. Springer.

Chen, Xi, Deng, Xiaotie, and Teng, Shang-Hua. 2009. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM (JACM)*, **56**(3), 1–57.

Chen, Xi, Kroer, Christian, and Kumar, Rachitesh. 2021a. The Complexity of Pacing for Second-Price Auctions. In: *Proceedings of the 2021 ACM Conference on Economics and Computation*.

Chen, Xi, Kroer, Christian, and Kumar, Rachitesh. 2021b. Throttling Equilibria in Auction Markets. Page 551 of: *Web and Internet Economics - 17th International Conference, WINE*. Lecture Notes in Computer Science, vol. 13112. Springer.

Cheung, Yun Kuen, Cole, Richard, and Rastogi, Ashish. 2012. Tatonnement in ongoing markets of complementary goods. Pages 337–354 of: *Proceedings of the 13th ACM Conference on Electronic Commerce*.

Cheung, Yun Kuen, Cole, Richard, and Devanur, Nikhil R. 2019. Tatonnement beyond gross substitutes? Gradient descent to the rescue. *Games and Economic Behavior*.

Cieliebak, Mark, Eidenbenz, Stephan J, Pagourtzis, Aris, and Schlude, Konrad. 2008. On the Complexity of Variations of Equal Sum Subsets. *Nord. J. Comput.*, **14**(3), 151–172.

Clarke, Edward H. 1971. Multipart pricing of public goods. *Public choice*, 17–33.

Codenotti, Bruno, Saberi, Amin, Varadarajan, Kasturi, and Ye, Yinyu. 2006. Leontief economies encode nonzero sum two-player games. Pages 659–667 of: *SODA*, vol. 6.

Cole, Richard, and Fleischer, Lisa. 2008. Fast-converging tatonnement algorithms for one-time and ongoing market problems. Pages 315–324 of: *Proceedings of the fortieth annual ACM symposium on Theory of computing*.

Cole, Richard, Devanur, Nikhil R, Gkatzelis, Vasilis, Jain, Kamal, Mai, Tung, Vazirani, Vijay V, and Yazdanbod, Sadra. 2017. Convex program duality, fisher markets, and Nash social welfare. In: *18th ACM Conference on Economics and Computation, EC 2017*. Association for Computing Machinery, Inc.

Conitzer, Vincent, and Sandholm, Tuomas. 2006. Computing the optimal strategy to commit to. Pages 82–90 of: *Proceedings of the 7th ACM conference on Electronic commerce*.

Conitzer, Vincent, and Sandholm, Tuomas. 2008. New complexity results about Nash equilibria. *Games and Economic Behavior*, **63**(2), 621–641.

Conitzer, Vincent, Kroer, Christian, Sodomka, Eric, and Stier-Moses, Nicolás E. 2018. Multiplicative Pacing Equilibria in Auction Markets. In: *International Conference on Web and Internet Economics*.

Conitzer, Vincent, Kroer, Christian, Panigrahi, Debmalya, Schrijvers, Okke, Sodomka, Eric, Stier-Moses, Nicolas E, and Wilkens, Chris. 2019. Pacing Equilibrium in First-Price Auction Markets. In: *Proceedings of the 2019 ACM Conference on Economics and Computation*. ACM.

Conitzer, Vincent, Kroer, Christian, Sodomka, Eric, and Stier-Moses, Nicolas E. 2022a. Multiplicative pacing equilibria in auction markets. *Operations Research*, **70**(2), 963–989.

Conitzer, Vincent, Kroer, Christian, Panigrahi, Debmalya, Schrijvers, Okke, Stier-Moses, Nicolas E, Sodomka, Eric, and Wilkens, Christopher A. 2022b. Pacing equilibrium in first price auction markets. *Management Science*, **68**(12), 8515–8535.

Daskalakis, Constantinos, Goldberg, Paul W, and Papadimitriou, Christos H. 2009. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, **39**(1), 195–259.

Daskalakis, Constantinos, Deckelbaum, Alan, and Kim, Anthony. 2015. Near-optimal no-regret algorithms for zero-sum games. *Games and Economic Behavior*, **92**, 327–348.

Debreu, Gerard. 1952. A social equilibrium existence theorem. *Proceedings of the National Academy of Sciences*, **38**(10), 886–893.

Deligkas, Argyrios, Fearnley, John, Hollender, Alexandros, and Melissourgos, Themistoklis. 2024. Pure-circuit: Tight inapproximability for PPAD. *Journal of the ACM*, **71**(5), 1–48.

Devanur, Nikhil R, and Mehta, Aranyak. 2023. Online Matching in Advertisement Auctions. *Online and Matching-Based Market Design*, 130.

Devanur, Nikhil R, Jain, Kamal, Sivan, Balasubramanian, and Wilkens, Christopher A. 2019. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. *Journal of the ACM (JACM)*, **66**(1), 1–41.

Dwork, Cynthia, Hardt, Moritz, Pitassi, Toniann, Reingold, Omer, and Zemel, Richard. 2012. Fairness through awareness. Pages 214–226 of: *Proceedings of the 3rd innovations in theoretical computer science conference*. ACM.

Easley, David, Kleinberg, Jon, et al. 2010. *Networks, crowds, and markets: Reasoning about a highly connected world*. Vol. 1. Cambridge university press.

Edelman, Benjamin, and Ostrovsky, Michael. 2007. Strategic bidder behavior in sponsored search auctions. *Decision support systems*, **43**(1), 192–198.

Edelman, Benjamin, Ostrovsky, Michael, and Schwarz, Michael. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review*, **97**(1), 242–259.

Eilenberg, Samuel, and Montgomery, Deane. 1946. Fixed point theorems for multi-valued transformations. *American Journal of mathematics*, **68**(2), 214–222.

Eisenberg, Edmund. 1961. Aggregation of utility functions. *Management Science*, **7**(4), 337–350.

Eisenberg, Edmund, and Gale, David. 1959. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics*, **30**(1), 165–168.

Fan, Ky. 1952. Fixed-point and minimax theorems in locally convex topological linear spaces. *Proceedings of the National Academy of Sciences of the United States of America*, **38**(2), 121.

Fan, Zhiyuan, Kroer, Christian, and Farina, Gabriele. 2024. On the Optimality of Dilated Entropy and Lower Bounds for Online Learning in Extensive-Form Games. In: *Advances in Neural Information Processing Systems, NeurIPS 2024*.

Fang, Fei, Stone, Peter, and Tambe, Milind. 2015. When security games go green: Designing defender strategies to prevent poaching and illegal fishing. In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2017. Regret minimization in behaviorally-constrained zero-sum games. Pages 1107–1116 of: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

Farina, Gabriele, Ling, Chun Kai, Fang, Fei, and Sandholm, Tuomas. 2019a. Correlation in Extensive-Form Games: Saddle-Point Formulation and Benchmarks. In: Wallach, Hanna M., Larochelle, Hugo, Beygelzimer, Alina, d'Alché-Buc, Florence, Fox, Emily B., and Garnett, Roman (eds), *Advances in Neural Information Processing Systems, NeurIPS 2019*.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2019b. Online convex optimization for sequential decision processes and extensive-form games. Pages 1917–1925 of: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2019c. Optimistic Regret Minimization for Extensive-Form Games via Dilated Distance-Generating Functions. Pages 5222–5232 of: *Advances in Neural Information Processing Systems*.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2020. Stochastic Regret Minimization in Extensive-Form Games. In: *International Conference on Machine Learning*. PMLR.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2021a. Better Regularization for Sequential Decision Spaces: Fast Convergence Rates for Nash, Correlated, and Team Equilibria. In: *Proceedings of the 2021 ACM Conference on Economics and Computation*.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2021b. Faster Game Solving via Predictive Blackwell Approachability: Connecting Regret Matching and Mirror Descent. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI.

Farina, Gabriele, Grand-Clément, Julien, Kroer, Christian, Lee, Chung-Wei, and Luo, Haipeng. 2023. Regret matching+:(in) stability and fast convergence in games. *Advances in Neural Information Processing Systems*, **36**, 61546–61572.

Farina, Gabriele, Kroer, Christian, and Sandholm, Tuomas. 2025. Better regularization for sequential decision spaces: Fast convergence rates for nash, correlated, and team equilibria. *Operations Research*.

Filos-Ratsikas, Aris, Hansen, Kristoffer Arnsfelt, Høgh, Kasper, and Hollender, Alexandros. 2024. PPAD-membership for problems with exact rational solutions: a general approach via convex optimization. Pages 1204–1215 of: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*.

Flajolet, Arthur, and Jaillet, Patrick. 2017. Real-time bidding with side information. Pages 5168–5178 of: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Flaspohler, Genevieve E, Orabona, Francesco, Cohen, Judah, Mouatadid, Soukayna, Oprescu, Miruna, Orenstein, Paulo, and Mackey, Lester. 2021. Online learning with optimism and delay. Pages 3363–3373 of: *International Conference on Machine Learning*. PMLR.

Fudenberg, Drew, and Tirole, Jean. 1991. *Game theory*. MIT press.

Ganzfried, Sam, and Sandholm, Tuomas. 2015. Endgame Solving in Large Imperfect-Information Games. Pages 37–45 of: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*.

Gao, Yuan, and Kroer, Christian. 2020. First-order methods for large-scale market equilibrium computation. *Advances in Neural Information Processing Systems*, **33**.

Gao, Yuan, Kroer, Christian, and Goldfarb, Donald. 2021a. Increasing Iterate Averaging for Solving Saddle-Point Problems. In: *Proceedings of the AAAI Conference on Artificial Intelligence*.

Gao, Yuan, Peysakhovich, Alex, and Kroer, Christian. 2021b. Online market equilibrium with application to fair division. *Advances in Neural Information Processing Systems*, **34**, 27305–27318.

Ghodsi, Mohammad, HajiAghayi, MohammadTaghi, Seddighin, Masoud, Seddighin, Saeed, and Yami, Hadi. 2018. Fair allocation of indivisible goods: Improvements and generalizations. Pages 539–556 of: *Proceedings of the 2018 ACM Conference on Economics and Computation*.

Gilboa, Itzhak, and Zemel, Eitan. 1989. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, **1**(1), 80–93.

Glicksberg, Irving L. 1952. A further generalization of the Kakutani fixed point theorem, with application to Nash equilibrium points. *Proceedings of the American Mathematical Society*, **3**(1), 170–174.

Goldman, Jonathan, and Procaccia, Ariel D. 2015. Spliddit: Unleashing fair division algorithms. *ACM SIGecom Exchanges*, **13**(2), 41–46.

Grand-Clément, Julien, and Kroer, Christian. 2024. Solving optimization problems with Blackwell approachability. *Mathematics of Operations Research*, **49**(2), 697–728.

Gribik, Paul R, Hogan, William W, Pope, Susan L, et al. 2007. Market-clearing electricity prices and energy uplift. *Cambridge, MA*, 1–46.

Groves, Theodore. 1973. Incentives in teams. *Econometrica: Journal of the Econometric Society*, 617–631.

Gupta, Anupam, and Molinaro, Marco. 2016. How the experts algorithm can help solve lps online. *Mathematics of Operations Research*, **41**(4), 1404–1431.

Hart, Sergiu, and Mas-Colell, Andreu. 2000. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, **68**(5), 1127–1150.

Hazan, Elad, et al. 2016. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, **2**(3-4), 157–325.

Hoda, Samid, Gilpin, Andrew, Pena, Javier, and Sandholm, Tuomas. 2010. Smoothing techniques for computing Nash equilibria of sequential games. *Mathematics of Operations Research*, **35**(2), 494–512.

Hogan, William W. 1992. Contract networks for electric power transmission. *Journal of regulatory economics*, **4**(3), 211–242.

Jalota, Devansh, Pavone, Marco, Qi, Qi, and Ye, Yinyu. 2023. Fisher markets with linear constraints: Equilibrium properties and efficient distributed algorithms. *Games and Economic Behavior*, **141**, 223–260.

Jiang, Jiashuo, Li, Xiaocheng, and Zhang, Jiawei. 2025. Online stochastic optimization with wasserstein-based nonstationarity. *Management Science*.

Johari, Ramesh. 2007. *MS&E 336: Dynamics and Learning in Games*.

Kiekintveld, Christopher, Jain, Manish, Tsai, Jason, Pita, James, Ordóñez, Fernando, and Tambe, Milind. 2009. Computing optimal randomized resource allocations for massive security games. Pages 689–696 of: *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*.

Kirschen, Daniel S, and Strbac, Goran. 2018. *Fundamentals of power system economics*. John Wiley & Sons.

Kjeldsen, Tinne Hoff. 2001. John von Neumann's conception of the minimax theorem: a journey through different mathematical contexts. *Archive for history of exact sciences*, **56**(1), 39–68.

Koller, Daphne, Megiddo, Nimrod, and von Stengel, Bernhard. 1996. Efficient computation of equilibria for extensive two-person games. *Games and economic behavior*, **14**(2), 247–259.

Korzhyk, Dmytro, Conitzer, Vincent, and Parr, Ronald. 2010. Complexity of computing optimal stackelberg strategies in security resource allocation games. In: *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Krishna, Vijay. 2009. *Auction theory*. Academic press.

Kroer, Christian, and Peysakhovich, Alexander. 2019. Scalable Fair Division for'At Most One'Preferences. *arXiv preprint arXiv:1909.10925*.

Kroer, Christian, Peysakhovich, Alexander, Sodomka, Eric, and Stier-Moses, Nicolas E. 2019. Computing large market equilibria using abstractions. Pages 745–746 of: *Proceedings of the 2019 ACM Conference on Economics and Computation*.

Kroer, Christian, Waugh, Kevin, Kılınç-Karzan, Fatma, and Sandholm, Tuomas. 2020. Faster algorithms for extensive-form game solving via improved smoothing functions. *Mathematical Programming*, 1–33.

Kuhn, HW. 2016. A SIMPLIFIED TWO-PERSON POKER. *Contributions to the Theory of Games, Volume I*, 97.

Kurokawa, David, Procaccia, Ariel D, and Wang, Junxing. 2018. Fair enough: Guaranteeing approximate maximin shares. *Journal of the ACM (JACM)*, **65**(2), 1–27.

Lanctot, Marc, Waugh, Kevin, Zinkevich, Martin, and Bowling, Michael. 2009. Monte Carlo sampling for regret minimization in extensive games. Pages 1078–1086 of: *Advances in neural information processing systems*.

Lee, Chung-Wei, Kroer, Christian, and Luo, Haipeng. 2021. Last-iterate Convergence in Extensive-Form Games. In: *Advances in Neural Information Processing Systems, NeurIPS 2019*.

Lee, Euiwoong. 2017. APX-hardness of maximizing Nash social welfare with indivisible items. *Information Processing Letters*, **122**, 17–20.

Levin, Dave, LaCurts, Katrina, Spring, Neil, and Bhattacharjee, Bobby. 2008. Bittorrent is an auction: analyzing and improving bittorrent's incentives. Pages 243–254 of: *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*.

Liao, Luofeng, Gao, Yuan, and Kroer, Christian. 2022. Nonstationary dual averaging and online fair allocation. *Advances in Neural Information Processing Systems*, **35**, 37159–37172.

Lipton, Richard J, Markakis, Evangelos, Mossel, Elchanan, and Saberi, Amin. 2004. On approximately fair allocations of indivisible goods. Pages 125–131 of: *Proceedings of the 5th ACM Conference on Electronic Commerce*.

Liu, Mingyang, Farina, Gabriele, and Ozdaglar, Asuman. 2024. LiteEFG: An Efficient Python Library for Solving Extensive-form Games. *arXiv preprint arXiv:2407.20351*.

Mas-Colell, Andreu, Whinston, Michael Dennis, Green, Jerry R, et al. 1995. *Microeconomic theory*. Vol. 1. Oxford university press New York.

Moravcik, Matej, Schmid, Martin, Ha, Karel, Hladik, Milan, and Gaukrodger, Stephen J. 2016. Refining subgames in large imperfect information games. In: *Thirtieth AAAI Conference on Artificial Intelligence*.

Moravčík, Matej, Schmid, Martin, Burch, Neil, Lisỳ, Viliam, Morrill, Dustin, Bard, Nolan, Davis, Trevor, Waugh, Kevin, Johanson, Michael, and Bowling, Michael. 2017. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, **356**(6337), 508–513.

Nan, Tianlong, Gao, Yuan, and Kroer, Christian. 2024. On the Convergence of T\^ atonnement for Linear Fisher Markets. *arXiv preprint arXiv:2406.12526*.

Nash Jr, John F. 1950. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, **36**(1), 48–49.

Neller, Todd W, and Lanctot, Marc. 2013. An introduction to counterfactual regret minimization. In: *Proceedings of model AI assignments, the fourth symposium on educational advances in artificial intelligence (EAAI-2013)*, vol. 11.

Nemirovski, Arkadi. 2004. Prox-method with rate of convergence O (1/t) for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, **15**(1), 229–251.

Nemirovsky, Arkadi, and Yudin, David Borisovich. 1983. Problem complexity and method efficiency in optimization.

Nesterov, Yu. 2005a. Excessive gap technique in nonsmooth convex minimization. *SIAM Journal on Optimization*, **16**(1), 235–249.

Nesterov, Yu. 2005b. Smooth minimization of non-smooth functions. *Mathematical programming*, **103**, 127–152.

Nesterov, Yurii. 2009. Primal-dual subgradient methods for convex problems. *Mathematical programming*, **120**(1), 221–259.

Nesterov, Yurii, and Shikhman, Vladimir. 2018. Computation of Fisher–Gale Equilibrium by Auction. *Journal of the Operations Research Society of China*, **6**(3), 349–389.

Neumann, Von. 1937. Uber ein okonomsiches gleichungssystem und eine verallgemeinering des browerschen fixpunktsatzes. Pages 73–83 of: *Erge. Math. Kolloq.*, vol. 8.

Nisan, Noam, Roughgarden, Tim, Tardos, Eva, and Vazirani, Vijay V. 2007. *Algorithmic game theory*. Cambridge University Press.

Ok, Efe A. 2011. *Real analysis with economic applications*. Princeton University Press.

O'Neill, Richard P, Sotkiewicz, Paul M, Hobbs, Benjamin F, Rothkopf, Michael H, and Stewart Jr, William R. 2005. Efficient market-clearing prices in markets with nonconvexities. *European journal of operational research*, **164**(1), 269–285.

Orabona, Francesco. 2019. A Modern Introduction to Online Learning. *arXiv preprint arXiv:1912.13213*.

Osborne, Martin J, and Rubinstein, Ariel. 1994. *A course in game theory*. MIT press.

Othman, Abraham, Papadimitriou, Christos, and Rubinstein, Aviad. 2016. The complexity of fairness through equilibrium. *ACM Transactions on Economics and Computation (TEAC)*, **4**(4), 1–19.

Peysakhovich, Alexander, Kroer, Christian, and Usunier, Nicolas. 2023. Implementing Fairness Constraints in Markets Using Taxes and Subsidies. Pages 916–930 of: *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*.

Pita, James, Jain, Manish, Marecki, Janusz, Ordóñez, Fernando, Portway, Christopher, Tambe, Milind, Western, Craig, Paruchuri, Praveen, and Kraus, Sarit. 2008. Deployed ARMOR protection: the application of a game theoretic model for security at the Los Angeles International Airport.

Popov, Leonid Denisovich. 1980. A modification of the Arrow-Hurwicz method for search of saddle points. *Mathematical notes of the Academy of Sciences of the USSR*, **28**(5), 845–848.

Rakhlin, Alexander, and Sridharan, Karthik. 2013. Optimization, learning, and games with predictable sequences. Pages 3066–3074 of: *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*.

Romanovskii, IV. 1962. Reduction of a Game with Full Memory to a Matrix Game. *Doklady Akademii Nauk SSSR*, **144**(1), 62–+.

Ross, Sheldon M. 1995. *Stochastic processes*. John Wiley & Sons.

Roughgarden, Tim. 2016. *Twenty lectures on algorithmic game theory*. Cambridge University Press.

Sandberg, Sheryl. 2019. Doing More to Protect Against Discrimination in Housing, Employment and Credit Advertising. *Facebook*.

Schilling, René L. 2017. *Measures, integrals and martingales*. Cambridge University Press.

Schweppe, FC, Tabors, RD, Caraminis, MC, and Bohn, RE. 1988. Spot pricing of electricity.

Shmyrev, Vadim I. 2009. An algorithm for finding equilibrium in the linear exchange model with fixed budgets. *Journal of Applied and Industrial Mathematics*, **3**(4), 505.

Shoham, Yoav, and Leyton-Brown, Kevin. 2008. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press.

Sinha, Arunesh, Fang, Fei, An, Bo, Kiekintveld, Christopher, and Tambe, Milind. 2018. Stackelberg security games: Looking beyond a decade of success. IJCAI.

Sion, Maurice, et al. 1958. On general minimax theorems. *Pacific Journal of mathematics*, **8**(1), 171–176.

Southey, Finnegan, Bowling, Michael, Larson, Bryce, Piccione, Carmelo, Burch, Neil, Billings, Darse, and Rayner, Chris. 2005. Bayes' bluff: opponent modelling in poker. Pages 550–558 of: *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*.

Steele, J Michael. 2004. *The Cauchy-Schwarz master class: an introduction to the art of mathematical inequalities*. Cambridge University Press.

Stott, Brian, Jardim, Jorge, and Alsaç, Ongun. 2009. DC power flow revisited. *IEEE Transactions on Power Systems*, **24**(3), 1290–1300.

Stradi, Francesco Emanuele, Castiglioni, Matteo, Marchesi, Alberto, Gatti, Nicola, and Kroer, Christian. 2025. *No-Regret Learning Under Adversarial Resource Constraints: A Spending Plan Is All You Need!*

Sundaram, Rangarajan K. 1996. *A first course in optimization theory*. Cambridge university press.

Sweeney, James L. 2013. *The California electricity crisis*. Hoover Press.

Syrgkanis, Vasilis, Agarwal, Alekh, Luo, Haipeng, and Schapire, Robert E. 2015. Fast convergence of regularized learning in games. Pages 2989–2997 of: *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*.

Talluri, Kalyan, and Van Ryzin, Garrett. 1998. An analysis of bid-price controls for network revenue management. *Management science*, **44**(11-part-1), 1577–1593.

Tambe, Milind. 2011. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge university press.

Tammelin, Oskari. 2014. Solving large imperfect information games using CFR+. *arXiv preprint arXiv:1407.5042*.

Tammelin, Oskari, Burch, Neil, Johanson, Michael, and Bowling, Michael. 2015. Solving heads-up limit Texas Hold'em. In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Taylor, Joshua Adam. 2015. *Convex optimization of power systems*. Cambridge University Press.

Udell, Madeleine, and Townsend, Alex. 2019. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, **1**(1), 144–160.

Udell, Madeleine, Horn, Corinne, Zadeh, Reza, Boyd, Stephen, et al. 2016. Generalized low rank models. *Foundations and Trends® in Machine Learning*, **9**(1), 1–118.

Varian, Hal R. 2007. Position auctions. *international Journal of industrial Organization*, **25**(6), 1163–1178.

Varian, Hal R, and Harris, Christopher. 2014. The VCG auction in theory and practice. *American Economic Review*, **104**(5), 442–45.

Vickrey, William. 1961. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, **16**(1), 8–37.

von Neumann, John. 1928. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, **100**(1), 295–320.

von Neumann, John. 1959. On the theory of games of strategy. *Contributions to the Theory of Games*, **4**, 13–42.

von Stackelberg, Heinrich. 1934. *Marktform und gleichgewicht*. J. springer.

von Stengel, Bernhard. 1996. Efficient computation of behavior strategies. *Games and Economic Behavior*, **14**(2), 220–246.

von Stengel, Bernhard, and Zamir, Shmuel. 2010. Leadership games with convex strategy sets. *Games and Economic Behavior*, **69**(2), 446–457.

Wang, Qian, Yang, Zongjun, Deng, Xiaotie, and Kong, Yuqing. 2023. Learning to bid in repeated first-price auctions with budgets. Pages 36494–36513 of: *International Conference on Machine Learning*. PMLR.

Wood, Allen J, Wollenberg, Bruce F, and Sheblé, Gerald B. 2013. *Power generation, operation, and control*. John wiley & sons.

Wu, Fang, and Zhang, Li. 2007. Proportional response dynamics leads to market equilibrium. Pages 354–363 of: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*.

Yang, Zongjun, Liao, Luofeng, and Kroer, Christian. 2024. Greedy-Based Online Fair Allocation with Adversarial Input: Enabling Best-of-Many-Worlds Guarantees. Pages 9960–9968 of: *Proceedings of the 38th AAAI Conference on Artificial Intelligence (AAAI)*.

Young, H Peyton. 2004. *Strategic learning and its limits*. OUP Oxford.

Zinkevich, Martin, Johanson, Michael, Bowling, Michael, and Piccione, Carmelo. 2007. Regret minimization in games with incomplete information. Pages 1729–1736 of: *Advances in neural information processing systems*.

Index